



Perpustakaan SKTM

LATIHAN ILMIAH I (WXES3181)

**VHDL IMPLEMENTATION FOR JAWI  
CHARACTER RECOGNITION**  
IMPLEMENTASI VHDL UNTUK PENGECAMAN  
AKSARA JAWI

DISEDIAKAN OLEH  
**RAZMAN BIN ABDUL GHANI**  
**810929-05-5129**  
**WEK 000321**

SARJANA MUDA SAINS KOMPUTER  
(SISTEM DAN TEKNOLOGI RANGKAIAN)

DISEDIAKAN UNTUK  
**ENCIK ZAIDI BIN RAZAK (PENASIHAT)**  
**EN MOHD YAMANI IDNA IDRIS (MODERATOR)**

## ABSTRAK

Kajian terhadap pembangunan tulisan jawi di era teknologi maklumat yang canggih ini memang kian giat diusahakan pada masa kini kerana beberapa perisian yang berkaitan penulisan jawi ini yang telah diperkenalkan. Namun sememangnya kita ketahui, pengetahuan tentang tulisan jawi agak kurang didedahkan kepada masyarakat pada masa ini. Sumber yang biasa kita jumpa untuk mengenali tulisan jawi ini yang popular ialah akhbar harian jawi, itupun dalam kuantiti yang agak kecil. Contoh perisian yang membangunkan penulisan jawi ialah Jawi Pro Ofis 2002.

Jadi pada kajian ini, saya dan kumpulan projek telah berusaha untuk melaksanakan satu kajian yang bertujuan untuk mengecamkan aksara tulisan jawi yang kurang jelas seperti manuskrip lama atau tulisan tangan kepada imej yang lebih jelas. Beberapa proses telah dilakukan dalam melaksanakan projek ini. Dalam menjayakan teori kajian ini, algoritma yang digunakan adalah berdasarkan kaedah kod rantai. Selain itu kajian ini menggunakan konsep kejiranan Piksel sebagai satu langkah yang sesuai bagi membaca nilai imej pada piksel tersebut.

Diharapkan kajian ini dapat memberikan manfaat kepada semua lapisan masyarakat dan memperkembangkan lagi penulisan jawi ini kepada dunia luar. Semoga kajian pengecaman aksara tulisan jawi yang berimplementasikan bahasa perihalan perkakasan VHDL dapat direalisasikan dengan jayanya seterusnya membuat anjakan paradigma terhadap kemajuan penulisan jawi ini.



## PENGHARGAAN



Segala puji-pujian kepada Allah S.W.T . Selawat dan salam ke atas Nabi Muhammad S.A.W , keluarga serta sahabat-sahabat baginda sekalian. Syukur ke hadrat Ilahi kerana dengan izinnya dapat saya menyiapkan satu kajian “*Implementasi VHDI dalam pengecaman aksara jawi*” bagi memenuhi syarat untuk matapelajaran Projek Ilmiah Tahap Akhir 1 (WXES 3181) dapat disempurnakan dalam masa yang ditetapkan.

Setinggi penghargaan dan terima kasih kepada penasihat saya, Encik Zaidi Razak yang banyak membantu, memberi tunjuk ajar, sokongan dan motivasi yang tinggi kepada saya dalam memahami dengan lebih jelas kajian sedang saya jalankan ini. Tidak dilupakan juga kepada Encik Mohd. Yamani Idna Idris yang bertindak sebagai moderator dalam menjayakan kajian ini dan memberi panduan untuk memperbaiki teori yang kami rancang.

Selain itu, saya juga ingin mengambil kesempatan untuk mengucapkan jutaan terima kasih kepada rakan sekumpulan yang sama-sama menjalankan projek ini. Mereka yang dimaksudkan ialah **Nur Hasinah Hassan, Nor Nita Adila Mustafa, Suzylarina Samsudin , Nurainil Mustapha dan Hazrin Ahmad**. Komitmen, tunjuk ajar, saling membantu dan gabungan idea adalah intipati dalam memudahkan lagi pelaksanaan projek kami ini . Terima kasih juga diucapkan kepada semua yang terlibat secara tidak langsung dalam menjayakan projek ini.

Akhir sekali diharapkan kajian ini dapat memberi manfaat kepada setiap lapisan masyarakat. Sekian terima kasih.

ISI KANDUNGAN

3.1 Pengamalan kepada metodologi sistem..... 13-15

Abstrak.....i

Penghargaan.....ii

Isi Kandungan.....iii-v

Senarai Rajah.....vi-vii

3.2.3 Pengecapan untuk aksara tambahan dan penerbitannya dengan aksara asal

1.0 Pengenalan.....21

1.1 Pengenalan.....1

1.2 Objektif.....2

1.3 Skop Projek.....3

1.4 Jadual Perancangan Projek.....4

5.0 Pembangunan Sistem

2.0 Kajian Literasi.....29-38

2.1 Pengenalan.....5

2.2 Kajian Pencaman Jawi Melalui Rekabentuk Perkakasan.....6-7

2.3 Pencaman Huruf.....8

2.4 Teknik Kod Rantaian.....8

2.5 Teknik Pengecaman bit.....9

2.6 Perwakilan Imej Digital.....10-11

2.6 Kesimpulan.....12



### 3.0 Metodologi Sistem

3.1 Pengenalan kepada metodologi sistem.....	13-15
3.2 Penggunaan Kaedah Kod Rantaian.....	16-17
3.2.1 Kaedah Hough Transform.....	18
3.2.2 Penentuan nilai titik bersambungan dengan menggunakan konsep kejiranan piksel ( Pixel Neighbourhood).....	19
3.2.3 Pengecaman untuk aksara tambahan dan pemadannya dengan aksara asas bagi mewakili hasil pengecaman dalam bentuk kod.....	20-21

### 4.0 Rekabentuk Sistem

4.1 Rekabentuk gambaran keseluruhan perjalanan proses secara ringkas.....	22-23
4.2 Rekabentuk bercirikan gambarajah input/ouput.....	24-25
4.3 Rekabentuk Berasaskan gambarajah blok.....	26-28

### 5.0 Pembangunan Sistem

5.1 Algoritma membaca Kod Rantaian melalui kaedah tatasusunan.....	29-38
5.2 Algoritma membaca Kod Rantaian berdasarkan kejiranan piksel yang telah ditetapkan.....	39-52

### 5.0 Pengujian Sistem

6.1 Pengujian algoritma membaca Kod Rantaian melalui Kaedah Tatasusunan...	53-57
6.2 Pengujian Algoritma pembacaan Kod Rantaian berdasarkan kejiranan piksel.....	58-63

6.0 Perbincangan Sistem	7
7.1 Perbincangan bagi algoritma pembacaan Kod Rantaian melalui kaedah tatasusunan.....	64-66
7.2 Perbincangan bagi algoritma membaca Kod Rantaian berdasarkan kejiranan piksel yang telah ditetapkan.....	67-68
7.3 Kesimpulan Dan Cadangan.....	69-72
8.0 Rujukan.....	74
Rajah 6	15
Rajah 8	17
Rajah 9	17
Rajah 10	19
Rajah 11	20
Rajah 12	22
Rajah 13	24
Rajah 3.1.1	30
Rajah 3.1.2	31
Rajah 3.1.3	32
Rajah 3.2.1	39
Rajah 3.2.2	40



## Senarai Jadual/Rajah

Jadual 1	36 huruf jawi yang dibahagikan kepada Berkumpulan dan Individu	7
Rajah 1	Kaedah Lapan arah Kod rantai	9
Rajah 2	Kaedah empat arah Kod Rantai	9
Rajah 3	Gabungan Entiti-entiti Rekabentuk Perkakasan Khas Huruf Jawi	10
Rajah 4	Kelas Satu	11
Rajah 5	Kelas Dua	14
Rajah 6	Kelas Tiga	15
Rajah 7	Kaedah Lapan arah Kod Rantai	16
Rajah 8	Perwakilan Kod Rantai	17
Rajah 9	Arah Kod Rantai	17
Rajah 10	Ilustrasi Konsep Kejiranan Pikel	19
Rajah 11	Kedudukan Titik-titik pada koordinat	20
Rajah 12	Gambarajah Aliran Perjalanan Proses Keseluruhan	22
Rajah 13	Gambarajah Blok proses Pengecaman Aksara Tulisan Jawi	26
Rajah 5.1.1	Cara pembacaan pikel bagi tatasusunan 4 x 4	30
Rajah 5.1.2	Contoh Nilai Binari 16 bit Input yang diterima	31
Rajah 5.1.3	Gambarajah arah Kod Rantai	32
Rajah 5.2.1	Cara pembacaan bagi setiap pikel ke pikel berdasarkan konsep kejiranan 3 x 3	39
Rajah 5.2.2	Contoh Nilai Binari 9 bit Input yang diterima	40

Rajah 6.1.3	Rajah bagi simulasi	57
Rajah 6.2.3	Rajah bagi simulasi	62
Rajah 7.3.2	Implementasi kaedah matrik tatasusunan bagi pembacaan arah kod rantai	70

## BAB 1

### Pengenalan



## 1.1 Pengenalan

Di dalam bab ini akan diterangkan serba sedikit berkaitan perancangan keseluruhan proses bagi projek ini. Proses pengimplementasian VHDL dalam pengecutan aktara jawi ini adalah sub proses untuk satu sistem yang kompleks. Sebagaimana yang kita ketahui, tujuan utama projek ini adalah untuk mengecut imej aktara tulisan jawi yang kurang jelas kepada imej yang lebih mudah dilihat dan dibaca serta kajian ini dilakukan secara bersempadan. Oleh itu, terdapat beberapa proses terlibat dalam melaksanakan setiap kerja yang diperlukan bagi menghasilkan output yang dikehendaki.

Antara proses-proses yang terlibat ialah proses pengurangan background (noise reduction) ialah proses dimana background imej akan dihapuskan untuk menghasilkan imej yang lebih jelas. Proses ini melibatkan penggunaan teknik thresholding. Setelah itu, proses peningkatan kualiti imej (Image Enhancement) proses bagi meningkatkan dan mengkilatkan imej. Proses pengsegmentasian (segmentation) ialah proses untuk memisahkan aktara tulisan jawi yang bersempadan kepada aktara tunggal. Dengan ini, kita akan memudahkan kerja bagi mengecut aktara jawi ini yang akan menjadi sub proses pengecutan aktara jawi (Recognition). Proses ini dilakukan dalam perisian imej digital.

Apabila imej aktara jawi ini telah dapat dipisahkan/dikecam, ia akan diwakili dengan nombor yang akan dipaparkan dalam pengkalan data bagi mengenalpasti identiti imej ini. Saya berharap rakan saya iaitu Hazrin Ahmad bertanggungjawab dalam tugas untuk pengecutan aktara tulisan jawi melalui penggunaan bahasa VHDL. Jadi ini adalah serba sedikit mengenai berkenaan perancangan proses. Keseluruhan projek ini bagi membolehkan pembaca mendapat sedikit gambaran awal.

## 1.1 Pengenalan

Di dalam bab ini akan diterangkan serba-sedikit berkaitan perjalanan keseluruhan proses bagi projek ini. Proses pengimplementasian VHDL dalam pengecaman aksara jawi ini adalah sub proses untuk satu sistem yang kompleks. Sebagaimana yang kita ketahui, tujuan utama projek ini adalah untuk mengecam imej aksara tulisan jawi yang kurang jelas kepada imej yang lebih mudah dilihat dan dibaca serta kajian ini dilakukan secara berkumpulan. Oleh itu, terdapat beberapa proses terlibat dalam melaksanakan setiap kerja yang diperlukan bagi menghasilkan output yang dikehendaki.

Antara proses-proses yang terlibat ialah proses pengurangan gangguan (noise Reduction) iaitu proses dimana membuang bentuk-bentuk yang tidak dikehendaki pada imej itu bagi menjadikan imej itu kepada bentuknya yang sebenar. Selepas itu, peningkatan kualiti imej (Image Enhancement) iaitu proses bagi menajamkan dan menjelaskan lagi imej yang agak kabur atau sukar dibaca. Proses pengsegmenan (segmentation) ialah proses untuk memecahkan aksara tulisan jawi yang bersambungan kepada aksara tunggal. Dengan ini ia akan memudahkan kerja bagi mengecam aksara jawi itu yang akan melalui proses pengecaman aksara jawi (Recognition). Proses ini dilakukan dalam perwakilan imej digital.

Apabila imej aksara jawi ini telah dapat dipastikan/dikecam, ia akan diwakili dengan nombor yang akan dipadankan dalam pangkalan data bagi mengenalpasti identiti imej itu. Saya bersama rakan saya iaitu Hazrin Ahmad bertanggungjawab dalam tugas untuk pengecaman aksara tulisan jawi melalui penggunaan aturcara VHDL. Jadi ini adalah serba-sedikit imbasan berkenaan perjalananan proses keseluruhan projek ini bagi membolehkan pembaca mendapat sedikit gambaran awal.



## 1.2 OBJEKTIF

Objektif projek yang saya laksanakan ini terbahagi kepada 2 objektif. Sebagaimana yang dinyatakan, projek ini adalah salah satu sub kepada satu sistem yang agak besar. Jadi objektif itu dibahagi kepada objektif berkumpulan dan individu.

### 1.2.1 Objektif Berkumpulan.

- 1) Mengecam imej tulisan tangan jawi dengan memaparkan sebagai aksara jawi tunggal.
- 2) Mengurangkan gangguan imej melalui proses 'Noise Reduction' dan meningkatkan kualiti imej melalui proses 'Image Enhancement'.
- 3) Melakukan proses pengsegmenan dan pepadanan aksara bagi melakukan pengecaman terhadap aksara tersebut.
- 4) Memudahkan pengenalpastian aksara jawi yang agak kabur dan kurang jelas, jadi ini akan dapat menterjemahkan tulisan jawi dengan lebih lancar.

### 1.2.2 Objektif Individu

- 1) Mengimplementasikan aturcara pengecaman aksara tulisan jawi dengan menggunakan bahasa perkakasan VHDL.
- 2) Mengenalpasti aksara tulisan jawi yang mempunyai ciri-cirinya yang tersendiri agar mudah dibaca.
- 3) Melaksanakan algorithma dalam proses pengecaman aksara tulisan imej jawi dengan kaedah kod rangkaian.

### 1.3 Skop Projek

Skop projek ini adalah lebih kepada proses dimana aksara jawi itu dikecam untuk mengenalpasti identiti aksara tersebut. Proses pendigitalan imej dilakukan dalam proses input output imej aksara tulisan jawi itu. Jadi kod-kod binari digunakan dalam proses pengecaman imej ini. Antara skop yang telah saya kenalpastikan ialah:

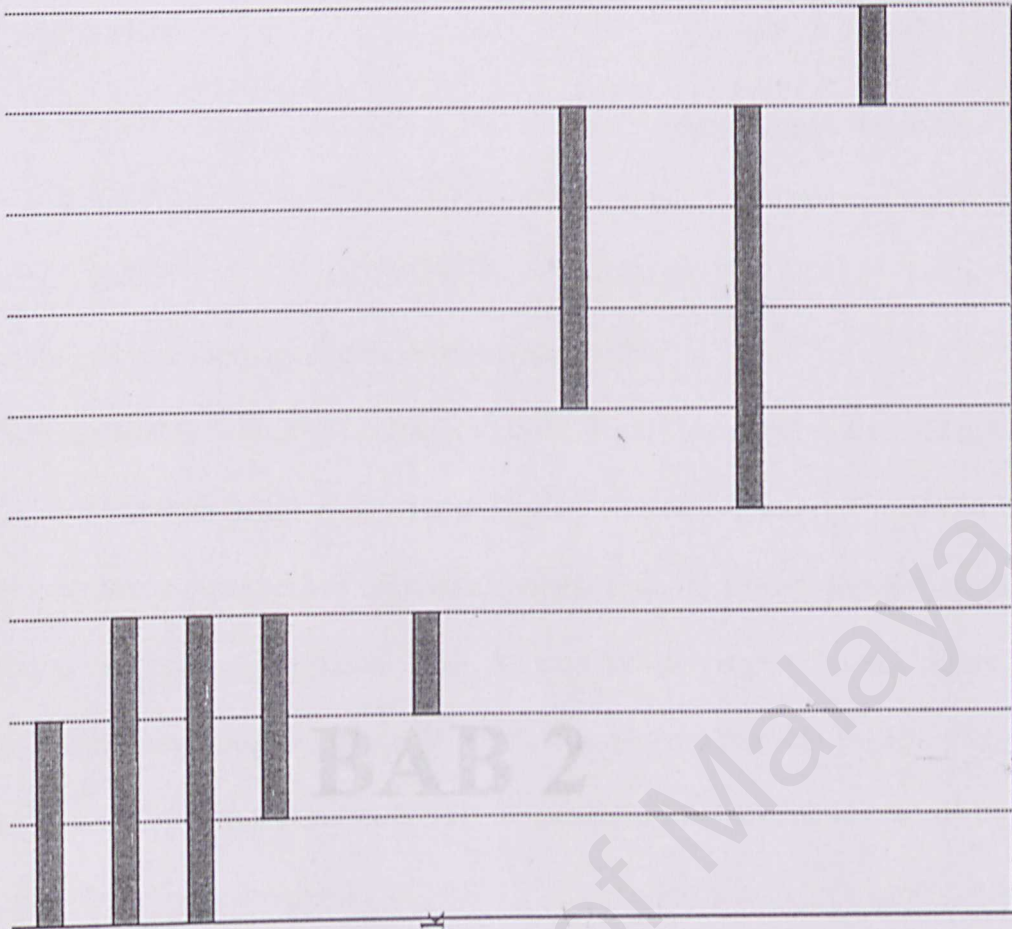
- 1) Algorithma kod rantai adalah kaedah utama yang diterapkan dalam melaksanakan proses pengecaman imej aksara jawi itu.
- 2) Keseluruhan perjalanan proses dilaksanakan dengan menggunakan bahasa perkakasan VHDL iaitu dalam bentuk kod-kod binari.
- 3) Saiz Bit yang digunakan dalam proses input dan output imej adalah 8 bit.
- 4) Pemadanan aksara asas dan aksara tambahan adalah berasaskan pangkalan data yang menyimpan identiti imej bagi setiap aksara dalam bentuk kod.
- 5) Output yang dikehendaki adalah aksara tunggal yang diwakilkan dalam bentuk kod.



1.4 Jadual Perancangan Projek

AKTIVITI:

- 1. Kajian dan pemahaman terhadap tajuk projek
- 2. Mempelajari Bahasa perihalan VHDL.
- 3. Memahami konsep Pengecaman imej secara teori.
- 4. Kajian terhadap metodologi/pendekatan yang digunakan.
- 5. Membuat Gambarajah Blok VHDL bagi Rekabentuk Sistem.
- 6. Merekabentuk pangkalan Data bagi proses proses pepadanan aksara tambahan + asas
- 7. Membina aturcara program untuk proses pengecaman.
- 8. Melaksanakan ujian terhadap proses yang telah dikaji.



JUN JULAI OGOS SEPT. OKT. NOV. DIS. JAN. FEB.

## 2.1 Pengenalan

Dalam menjalankan teori kajian ini, Bahasa perantara yang digunakan adalah VHDL. Ini kerana, kajian yang saya lakukan adalah pendigitalan imej dalam bahasa perkakasan. Jadi, VHDL adalah bahasa perihalan perkakasan yang sesuai diimplementasikan dalam algoritma proses pengesanan ini.

VHDL adalah akronim untuk (VHSIC Hardware Description Language). VHSIC adalah akronim untuk (Very High Speed Integrated Circuit). Ini adalah bahasa perihalan perkakasan yang boleh digunakan sebagai model sistem digital yang banyak tahap abstrak dan juga algoritma biasa kepada get-get logik. Sistem digital yang

## BAB 2

# KAJIAN LITERASI

Tulisan Jawi ini merupakan tulisan yang berstrukturkan aksara-aksara tulisan arab. Tulisan jawi ini juga adalah tulisan yang unik jika dilihat berbanding tulisan yang lain. Sebagai contoh, ia dibaca dari kanan ke kiri. Selain itu, proses pengesanan aksara tulisan jawi dengan menggunakan pengimplementasian bahasa perihalan perkakasan VHDL, ini adalah sistem yang agak baru pada masa ini. Banyak kajian berkenaan pengesanan tulisan jawi ini dilaksanakan dalam bentuk perisian. Dalam menjalankan perisian, kita melalui proses pengesanan aksara tulisan jawi ini, saya telah memahami bahasa hardware melalui kajian yang terdahulu.



## 2.1 Pengenalan

Dalam menjalankan teori kajian ini, Bahasa perantara yang digunakan adalah VHDL. Ini kerana, kajian yang saya lakukan adalah pendigitalan imej dalam bahasa perkakasan. Jadi, VHDL adalah bahasa perihalan perkakasan yang sesuai diimplementasikan dalam algoritma proses pengecaman ini.

VHDL adalah akronim untuk (VHSIC Hardware Description Language). VHSIC adalah akronim untuk (Very High Speed Integrated Circuit). Ini adalah bahasa perihalan perkakasan yang boleh digunakan sebagai model sistem digital pada banyak tahap abstrak dan juga algoritma biasa kepada get-get logik. Sistem digital yang kompleks boleh mengubahsuai daripada get-get logik yang ringkas kepada yang lebih kompleks. Sistem digital ini juga boleh diterangkan dalam bentuk hieraki. Jika kita dapat mengenali bahasa pengaturcaraan seperti C, C++ atau sebagainya sebagai bahasa pengantaraan atau 'enjin' bagi perisian-perisian tertentu, maka VHDL pula adalah bahasa pengantaraan bagi perkakasan.

Tulisan jawi ini merupakan tulisan yang berinspirasi aksara-akara tulisan arab. Tulisan jawi ini juga merupakan tulisan yang unik jika diteliti berbanding tulisan yang lain. Sebagai contohnya, ia dibaca dari kanan ke kiri. Selain itu, proses pengecaman aksara tulisan jawi dengan menggunakan pengimplementasian bahasa perihalan perkakasan VHDL ini adalah suatu yang agak baru pada masa kini. Banyak kajian berkenaan pengecaman tulisan jawi ini dilaksanakan dalam bentuk perisian. Dalam menjalankan penyelidikan terhadap proses pengecaman aksara tulisan jawi ini, saya telah membuat kajian literasi terhadap kajian yang terdahulu.

Kajian tersebut adalah:

## 2.2 Kajian Pencaman Jawi melalui Rekabentuk Perkakasan

Tajuk : Rekabentuk Perkakasan Khas Pencaman Jawi

Kajian oleh : Omar Abdul Rahim, Zaidi Razak, Noorzaily Mohammed Noor, Mashkuri

Yaakob. Fakulti Sains Komputer Dan teknologi Maklumat, Universiti

Malaya.

### 2.2.1 Metodologi yang digunakan

Kajian ini memfokuskan model perkakasan untuk mendigitalkan tulisan jawi manuskrip (tulisan jawi tangan). Output model ini ialah tulisan jawi yang boleh dibaca oleh sebarang pemproses kata yang ada font jawi. Kaedah yang digunakan ialah pengecaman pakej-pakej bentuk aksara serupa dan penganalisan titik aksara melalui teknik kod rangkaian. Model ini terbahagi kepada tiga modul iaitu pensegmenan aksara, pemprosesan imej imej (pembaikpulih, pengurangan kebisingan, pengecaman aksara) dan pencarian identiti aksara (melalui kod rangkaian).

Model ini memerlukan input satu perkataan jawi yang diimbis dan seterusnya dipecahkan ke dalam bentuk aksara-aksara tunggal dalam modul pemecahan aksara yang kemudiannya dikenalpasti aksara-aksara tersebut oleh modul pemproses aksara dan diberikan identitinya.

Modul akhir akan mencari aksara digital yang bersepadan dengan identity yang telah diberikan oleh modul pengecaman dalam jadual identiti aksara. Model ini akan direalisasikan dalam bentuk cip tunggal dengan menggunakan bahasa perihalan perkakasan VHDL pada peringkat seterusnya. Pendekatan yang digunakan ialah dengan



membahagikan huruf jawi yang terdiri daripada 36 huruf kepada bentuk berkumpulan dan bentuk individu seperti yang ditunjukkan dalam jadual 1. Bentuk berkumpulan boleh dibezakan diantara satu sama lain melalui bilangan titik dan lokasi titik-titiknya. Setiap huruf jawi mempunyai tiga jenis bentuk berlainan yang bersandarkan kepada kedudukannya dalam satu perkataan tulisan jawi. 3 jenis bentuk tersebut dikategorikan sebagai bentuk huruf diawal, ditengah dan diakhir dalam satu perkataan tulisan jawi. Bagi huruf-huruf jawi berkumpulan ia mempunyai bentuk yang sama samada diawal, ditengah mahupun diakhir yang membezakannya ialah titik-titiknya. Walaubagaimanapun terdapat huruf-huruf jawi yang mana bentuk tengah dan akhirnya sama seperti و, ز, ر, ذ, د, ا.

Jadual 1: Terdapat 36 huruf jawi yang dibahagikan kepada huruf-huruf

Berkumpulan dan individu

Bentuk Berkumpulan	Bentuk Individu
ث ت ب	ا
چ خ ج ح	ل
ذ	م
ز	ه
س ش	لا
ض ص	ء
ط ظ	ا
غ غ ع	
ق ق ف	
ک ک ی	
و	
ن ث	



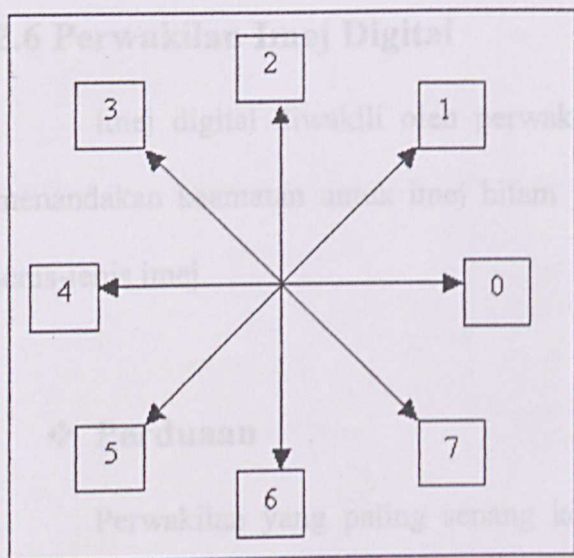
## 2.3 Pencaman Huruf

Pencaman huruf bermaksud sesuatu proses yang mengenal sesuatu huruf (tidak digital) dan seterusnya memaparkan huruf tersebut dalam keadaan yang lebih berkualiti (digital) daripada asal. Pencaman ini secara teorinya amat mudah untuk dilaksanakan tetapi pada pelaksanaannya ianya perlu melalui asas-asas pemprosesan imej secara digital seperti penyahbisingan, peningkatan kualiti dan seterusnya pensегmenan yang mana ianya melibatkan pemecahan ayat kepada huruf-huruf.

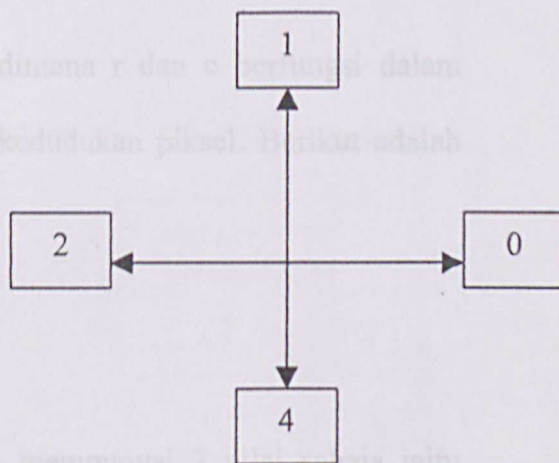
Kebanyakan penyelidikan yang dibuat adalah tertumpu kepada mencari identiti tenaga dalam setiap huruf yang ingin dicamkan. Dalam kertas ini proses-proses yang wujud dalam pemprosesan imej digital tidak akan diterangkan dan penerangan akan bertumpu kepada Algoritma Kod Rantaian yang akan digunakan dalam pencaman huruf.

## 2.4 Teknik Kod Rantaian

Kod rantaiian ialah jujukan nombor-nombor yang mewakili bentuk asas aksara-aksara berdasarkan penakrifan pada *gambarajah arah LETAK RUJUKAN*. Asas penomboran arah yang digunakan dalam kod rantaiian dibahagikan kepada 2 kaedah iaitu kaedah 8 arah kod rantaiian seperti yang ditunjukkan dalam Rajah 1 dan Kaedah 4 arah kod rantaiian bagi Rajah 4.



Rajah 1 : Kaedah lapan arah kod Rantaian



Rajah 2 : Kaedah empat arah kod rantai

Teknik kod rantai ini adalah bertujuan untuk menentukan jenis arah pergerakan imej bagi menetapkan nombor-nombor kod rantai yang akan mewakili imej tersebut. Ianya akan dilakukan pada bahagian atas dan tengah sahaja. Kod rantai akan diberikan mengikut arah jam walau dalam apa bentuk sekalipun dengan memeriksa kejiranan piksel tersebut.

## 2.5 Teknik pengecaman bit

Kaedah pengecaman bit ini adalah teknik yang digunakan bagi mengesan aksara tambahan. Kaedah menentukan samada terdapat titik adalah dengan mengenalpasti setiap koordinat yang mengandungi imej (dot) kerana ia bukan merupakan garisan bersambung atau mempunyai bentuk. Kaedah ini adalah mudah kerana titik tidak merupakan imej yang mempunyai bentuk samada lengkung, garisan dan sebagainya.



## 2.6 Perwakilan Imej Digital

Imej digital diwakili oleh perwakilan  $I(r,c)$  dimana  $r$  dan  $c$  berfungsi dalam menandakan keamatan untuk imej hitam putih bagi kedudukan piksel. Berikut adalah jenis-jenis imej.

### ❖ Perduaan

Perwakilan yang paling senang kerana ianya mempunyai 2 nilai sahaja iaitu hitam dan putih, '0' atau '1'. Ia selalu digunakan dalam pemproses imej pada komputer.

### ❖ Penskalaan kelabu

Perwakilan khas yang mengandungi maklumat mengenai keamatan imej. Bilangan bit yang ada menentukan tahap keamatan sesuatu imej. Contohnya 8 bit/piksel mengandungi tahap 256(0-255).

### ❖ Warna

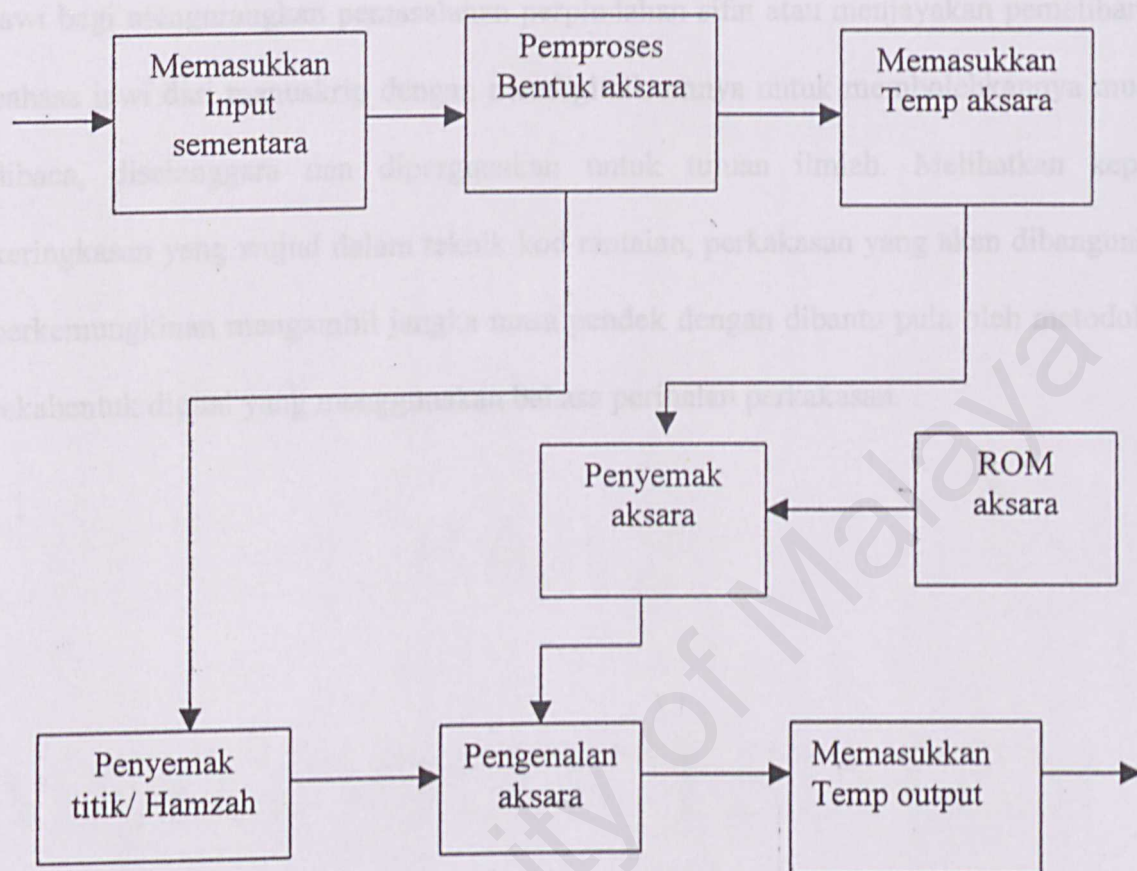
3-band monokrom imej yang mana setiap band mewakili warna yang berlainan. Sekiranya setiap band mempunyai 8 bit per piksel maka untuk piksel dalam perwakilan warna mempunyai 24 bit per piksel. Ini merupakan perwakilan RGB(red, blue, green).

### ❖ Pelbagai spektral

Maklumat yang disimpan menjangkau darjah penglihatan manusia. Contohnya seperti: X-ray dan kadar. Imej seperti ini selalunya ditukarkan kepada perwakilan seperti pemetaan kepada RGB.



Rajah 3 menunjukkan gabungan yang dicadangkan untuk Entiti-Entiti yang dinyatakan



Rajah 3 : Gabungan Entiti Rekabentuk Perkakasan Khas Huruf Jawi

Proses pengabungan ini akan dilakukan setelah semua Entiti-Entiti ini diuji ketepatan pemrosesannya.

## **2.7 Kesimpulan**

Kajian ini merupakan asas kepada pembangunan perkakasan pengecaman huruf jawi bagi mengurangkan pemasalahan perpindahan sifat atau menjayakan pemeliharaan bahasa jawi dari manuskrip dengan mendigitalkannya untuk membolehkannya mudah dibaca, diselenggara dan dipergunakan untuk tujuan ilmiah. Melihatkan kepada keringkasan yang wujud dalam teknik kod rantai, perkakasan yang akan dibangunkan berkemungkinan mengambil jangka masa pendek dengan dibantu pula oleh metodologi rekabentuk digital yang menggunakan bahasa perihalan perkakasan.

BAB 3

METODOLOGI SISTEM



### 3.1 PENGENALAN KEPADA METODOLOGI SISTEM

Dalam bahagian Metodologi sistem ini, saya telah menggunakan beberapa pendekatan dalam proses mengesan setiap imej aksara tulisan jawi ini. Sebagaimana yang kita ketahui, bilangan aksara tulisan jawi terdapat 36 huruf. Setiap aksara ini mempunyai bentuk-bentuk yang tersendiri. Secara tidak langsung, pastinya akan terdapat kesukaran dalam mengenalpasti identiti setiap aksara ini. Jadi saya telah membuatkan 3 pengelasan bagi memudahkan lagi proses pengesanan setiap aksara ini.

Pengelasan dilakukan berdasarkan bentuk dan aksara jawi ini sendiri. Aksara-aksara akan dikelaskan ke dalam beberapa kumpulan yang hampir sama. Contohnya huruf (ث ث ب) 'ta', 'te', 'da' mempunyai bentuk yang sama tetapi

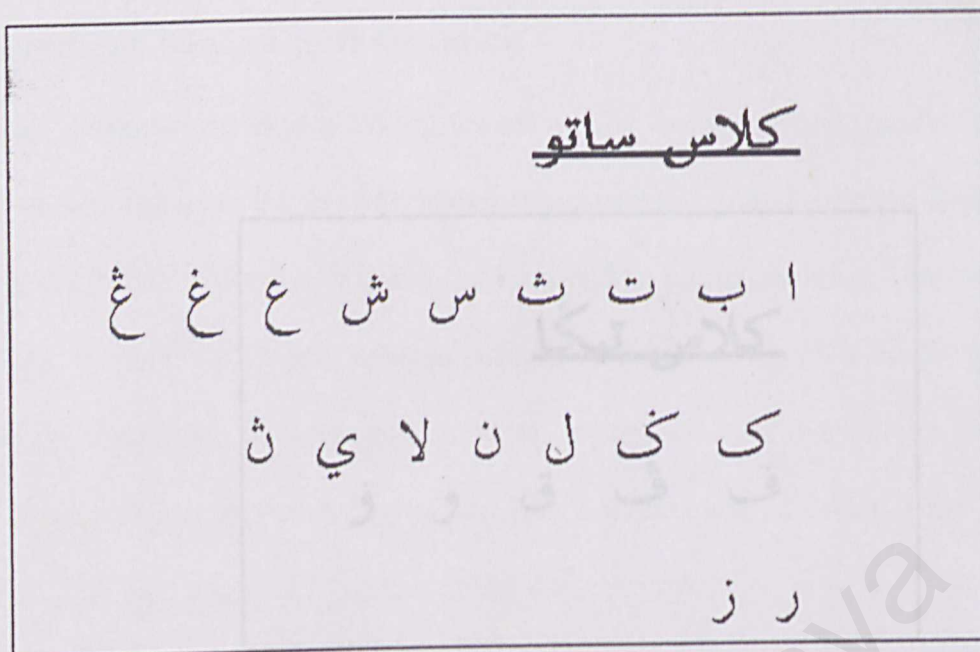
## BAB 3 METODOLOGI SISTEM

1, kelas 2 dan kelas 3.ajah 1, nahajah 3 menunjukkan kelas-kelas tersebut.

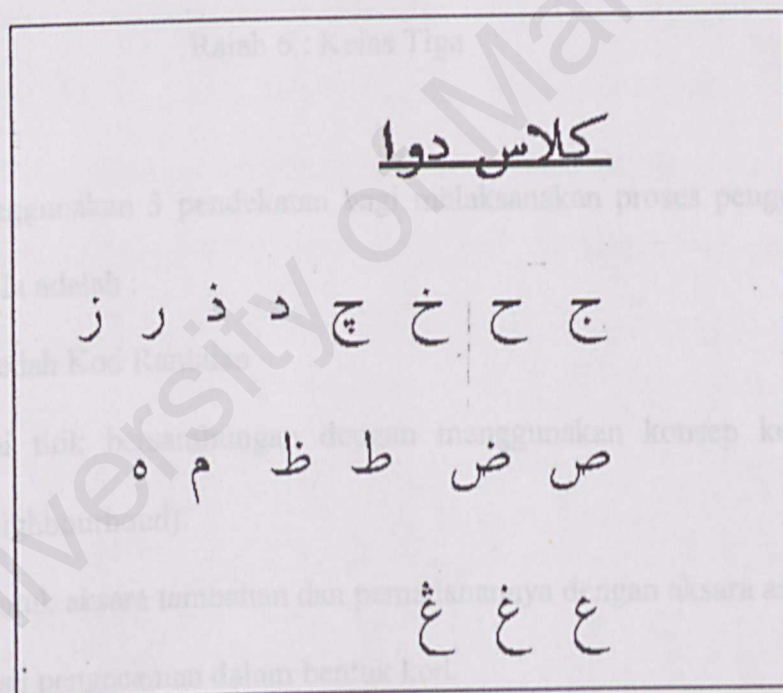
### 3.1 PENGENALAN KEPADA METODOLOGI SISTEM

Dalam bahagian Metodologi sistem ini, saya telah menggunakan beberapa pendekatan dalam proses mengecam setiap imej aksara tulisan jawi itu. Sebagaimana yang kita ketahui, bilangan aksara tulisan jawi terdapat 36 huruf. Setiap aksara itu mempunyai bentuk-bentuk yang tersendiri. Secara tidak langsung, pastinya akan terdapat kesukaran dalam mengenalpasti identiti setiap aksara itu. Jadi saya telah membuatkan 3 pengkelasan bagi memudahkan lagi proses pengecaman setiap aksara itu.

Pengkelasan dilakukan berasaskan bentuk dan saiz aksara jawi itu sendiri. Aksara-aksara akan dikelaskan berdasarkan bentuk-bentuknya yang hampir sama. Contohnya huruf (ث ت ب) 'ba', 'ta', 'tha' mempunyai bentuk yang sama tetapi jumlah titik yang berbeza, jadi inilah faktor yang membolehkan saya mengelaskan huruf-huruf ini didalam kelas yang sama. Kelas-kelas itu terbahagi kepada 3 iaitu kelas 1, kelas 2 dan kelas 3. rajah 1, rajah 2 dan rajah 3 menunjukkan kelas-kelas tersebut.



Rajah 4 : Kelas satu.

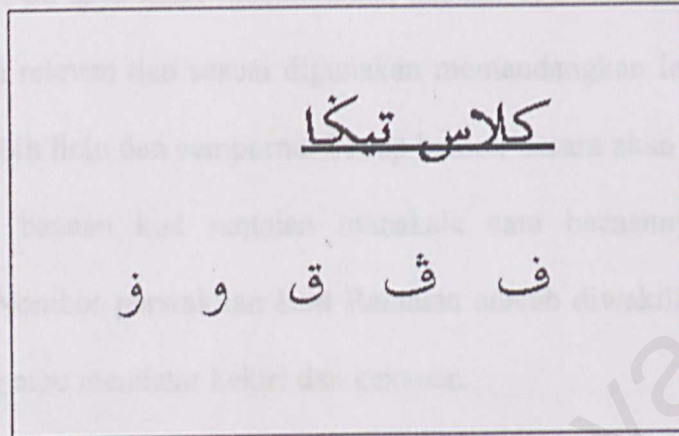


Rajah 5 : Kelas Dua



### 3.2 Penggunaan kaedah Kod Rantaian

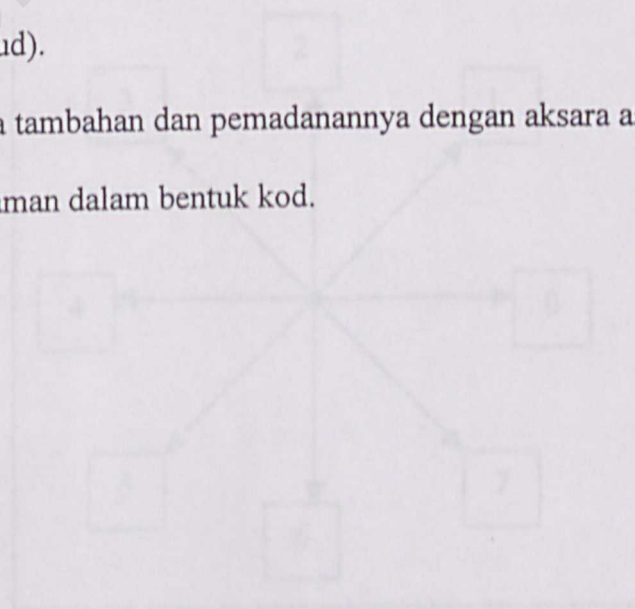
Pada proses pengiraan aksara-aksara ini saya telah menggunakan kaedah kod rantai. Jadi dalam bahagian ini akan saya tunjukkan bagaimana kaedah kod rantai ini lebih lanjut. Kaedah ini lebih relevan dan sesuai digunakan memandangkan ia yang diwakilinya akan kelihatan lebih jelas dan terperinci. Untuk itu, saya akan tunjukkan jawapan nombor berdasarkan kaedah kod rantai manakala cara bagaimana ia lebih mengikut arah putaran jam. Kemudian, saya akan tunjukkan bagaimana ia akan membentuk bentuk, mengikut kaedah kod rantai ini.



Rajah 6 : Kelas Tiga

Saya telah menggunakan 3 pendekatan bagi melaksanakan proses pengecaman aksara tulisan jawi ini. Ia adalah :

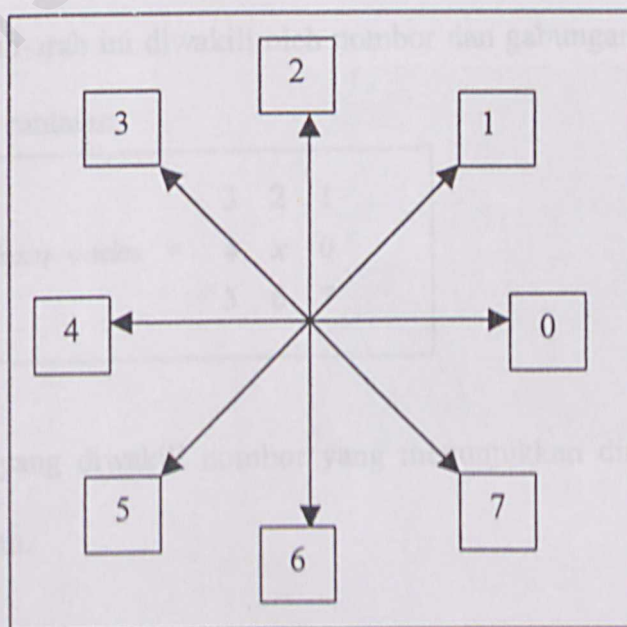
- 1) Penggunaan kaedah Kod Rantaian
- 2) Penentuan nilai titik bersambungan dengan menggunakan konsep kejiranan piksel (Pixel Neighbourhood).
- 3) Pengecaman untuk aksara tambahan dan pepadannya dengan aksara asas bagi mawakulkan hasil pengecaman dalam bentuk kod.



### 3.2 Penggunaan kaedah Kod Rantaian

Pada proses pengecaman aksara-aksara ini saya telah menggunakan kaedah kod rantaian. Jadi dalam bahagian ini saya akan menerangkan kaedah kod rantaian dengan lebih lanjut. Kaedah ini lebih relevan dan sesuai digunakan memandangkan imej yang diwakilinya akan kelihatan lebih licin dan sempurna. Setiap bentuk aksara akan diwakili jujukan nombor berdasarkan bacaan kod rantaian manakala cara bacaannya ialah mengikut arah putaran jam. Nombor perwakilan Kod Rantaian adalah diwakili samada bentuk menegak, melengkung atau mendatar kekiri dan kekanan.

Kaedah kod rantaian ini menggunakan pendekatan konsep '*Kejiranan Piksel*' iaitu kaedah yang membaca piksel yang mempunyai nilai secara bersambungan. Apabila imej diwakili oleh piksel-piksel, secara tidak langsung piksel-piksel ini juga akan mempunyai nilai. Piksel yang diwakili imej akan bernilai 1 manakala piksel yang tidak diwakili imej akan bernilai 0. Jadi, konsep '*kejiranan Piksel*' digunakan bagi membaca nilai 1 secara berturutan.



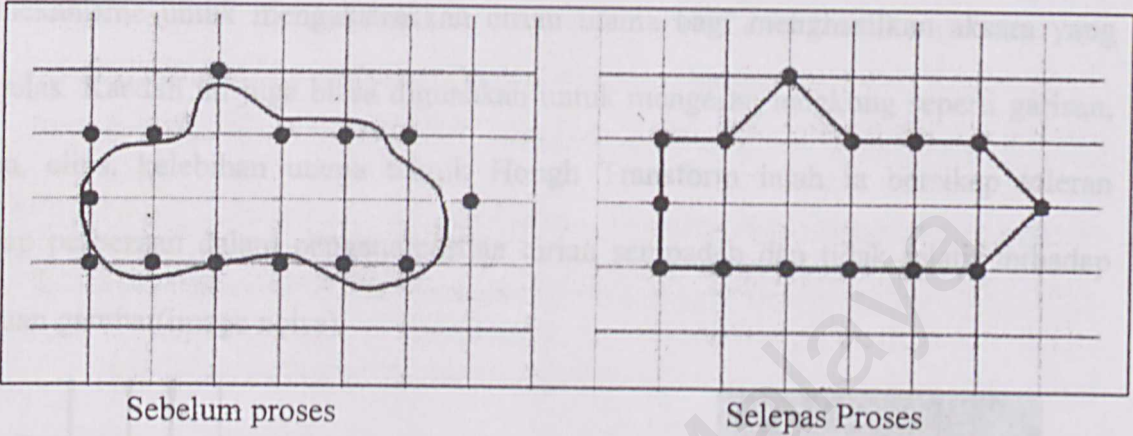
Rajah 7: Kaedah lapan arah kod rantaian.



3.2.1 Kaedah 'Hough Transform'

Kaedah Hough transform ini adalah untuk mengembalikan kedudukan imej yang

Rajah 7 menunjukkan gambarajah kod rantai yang mempunyai 8 arah.



Rajah 8 : Perwakilan kod rantai dalam pengecaman imej.

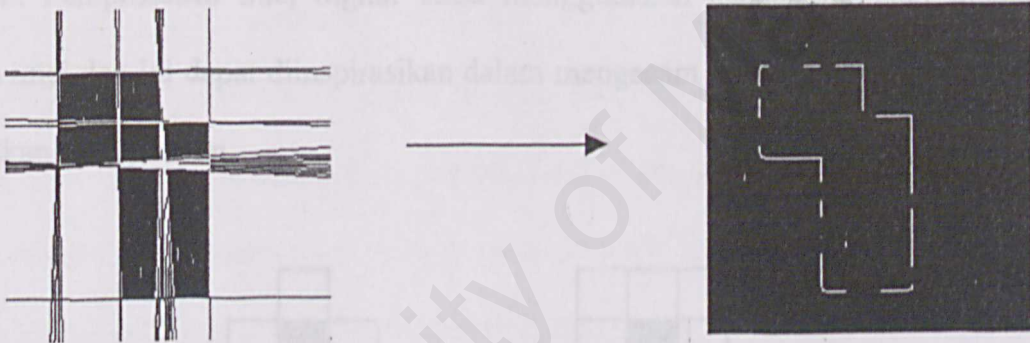
Rajah 8 menunjukkan bagaimana kedudukan imej diwakilkan dalam bentuk arah dan arah imej ini akan menghasilkan jujukan kod rantai bagi mengenalpasti indentiti imej. Kaedah ini diimplementasikan ketika mengecam aksara tulisan jawi nanti. Melalui kaedah arah inilah yang digunakan bagi membaca nilai piksel-piksel secara bersambungan. Setiap bentuk arah-arrah ini diwakili oleh nombor dan gabungan jujukan nombor ini akan membentuk kod rantai.

	3	2	1
Chain codes =	4	x	0
	5	6	7

Rajah 9 : Arah Kod Rantaian yang diwakili nombor yang menunjukkan dimana 'x' menjadi pusat bagi penentuan arah.

### 3.2.1 Kaedah 'Hough Transform'

Kaedah Hough transform ini adalah untuk mengembalikan kedudukan imej yang tidak berapa jelas kepada yang lebih jelas atau normal. Kaedah ini juga dikenali sebagai satu mekanisme untuk mengakstrakkan cirian utama bagi menghasilkan aksara yang lebih jelas. Kaedah ini juga biasa digunakan untuk mengesan lengkung seperti garisan, bulatan, elips, kelebihan utama teknik Hough Transform ialah ia bersikap toleran terhadap perbezaan dalam pengenaltastian cirian sempadan dan tidak relatif terhadap gangguan gambar(image noise).



Gambarajah di atas adalah aplikasi Hough Transform dimana ia mengesan kedudukan imej yang kurang jelas.



A



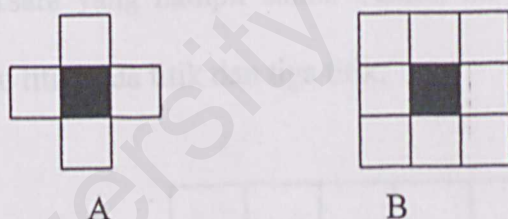
B



Gambarajah A di atas adalah menunjukkan sebuah gambar bangunan manakala Gambarajah B menunjukkan kedudukan garisan imej gambar bangunan itu. Ini adalah aplikasi yang digunakan dengan kaedah Hough Transform.

### 3.2.2 Penentuan nilai titik bersambungan dengan menggunakan konsep kejiranan piksel (Pixel Neighborhood)

Pendekatan dengan menggunakan kaedah konsep kejiranan adalah penting untuk dalam mengenalpasti identiti imej aksara tulisan jawi yang agak sama bentuk saiz dan cirinya. Konsep ini penting dalam menentukan kelas bagi aksara-aksara tulisan jawi tersebut.. Pemprosesan imej digital biasa menggunakan kaedah ini bagi mengenali sesuatu imej dan ini dapat diinspirasi dalam mengenali tulisan jawi yang juga boleh diwakilkan sebagai imej.



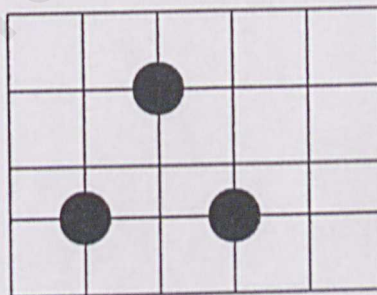
Rajah 10 : Ilustrasi Konsep Kejiranan Piksel

Rajah 10 menunjukkan gambaran proses kejiranan piksel. Gambarajah A adalah dipanggil 4-kejiranan iaitu piksel yang berlorek mempunyai nilai 1 dan dikelilingi 4 piksel yang bernilai 0. Gambarajah B pula dipanggil 8-kejiranan iaitu 1 piksel bernilai 1 dikelilingi 8 piksel bernilai 0. Jadi teori di praktikkan dengan mencari piksel kejiranan yang mempunyai nilai 1. Konsep ini juga dapat diterangkan melalui definisi region( keterhubungan kawasan) .

Bagi suatu region R dikatakan saling connected jika terdapat mana-mana 2 piksel  $(X_A, Y_A)$ ,  $(X_B, Y_B)$  milik R dapat dihubungkan oleh laluan  $(X_A, Y_A), \dots (X_{i-1}, Y_{i-1}), (X_i, Y_i), (X_{i+1}, Y_{i+1}), \dots (X_B, Y_B)$  di mana piksel  $(X_i, Y_i)$  milik R dan mana-mana piksel  $(X_i, Y_i)$  adalah berdekatan dengan piksel sebelumnya  $(X_{i-1}, Y_{i-1})$  dan piksel selepasnya  $(X_{i+1}, Y_{i+1})$  dalam laluan tersebut.

### 3.2.3 Pengecaman untuk aksara tambahan dan pepadannya dengan aksara asas bagi mawaklkan hasil pengecaman dalam bentuk kod.

Pengecaman untuk aksara tambahan dan pepadannya dengan aksara asas bagi mewakili hasil pengecaman dalam bentuk kod. Pendekatan ini mengesan aksara tambahan ini adalah berasaskan bilangan titik bagi setiap aksara tersebut. Ini akan dapat membezakan aksara-aksara yang hampir sama. Teknik ini bertumpu kepada titik-titik yang mungkin iaitu satu titik, dua titik dan tiga titik.



Rajah 11 : kedudukan titik-titik pada koordinat.

Rajah 11 menunjukkan teknik dimana untuk mengesan koordinat bagi titik-titik yang akan dikecamkan lokasinya dan bilangannya. Jadi logik pengecaman adalah seperti berikut:



Rajah 15 menunjukkan teknik dimana untuk mengesan koordinat bagi titik-titik yang akan dikecam kan lokasinya dan bilangannya. Jadi logik pengecaman adalah seperti berikut:

Titik 3: Terdapat 2 koordinat dengan paksi-x yang sama serta 1 koordinat pada paksi- y yang berlainan.

Titik 2: 2 koordinat dengan paksi-x yang sama.

Titik 1: 1 koordinat tunggal.

Apabila Nilai kod rangkaian bagi aksara tambahan dan juga nilai kod bagi aksara tambahan telah asas telah dikenali, proses pemadanan gabungan kod aksara tambahan dan aksara asas itu akan dirujuk pada pangkalan data yang telah dibuat. Ini bagi menghasilkan output aksara jawi tunggal yang telah dikecam dalam perwakilan kod. Jadi, ini adalah beberapa pendekatan yang telah digunakan bagi melaksanakan proses pengecaman terhadap aksara-aksara jawi tersebut.

## 4.0 REKABENTUK SISTEM

Fasa Rekabentuk Sistem dilaksanakan untuk membangunkan suatu rekabentuk fizikal berdasarkan rekabentuk logik sistem dengan menimbangkan keperluan yang telah ditetapkan dalam fasa analisis sistem.

Aktiviti-aktiviti rekabentuk sistem boleh dibahagikan kepada tiga bahagian :

- A. Rekabentuk gambaran keseluruhan perjalanan proses secara ringkas
- B. Rekabentuk berisikan gambaran input/output
- C. Rekabentuk berdasarkan Gambar rajai blok.

### 4.1 Rekabentuk Gambaran Perjalanan Proses Secara Ringkas

## REKABENTUK SISTEM



Rajah 12 : Gambaran alirah perjalanan proses keseluruhan.



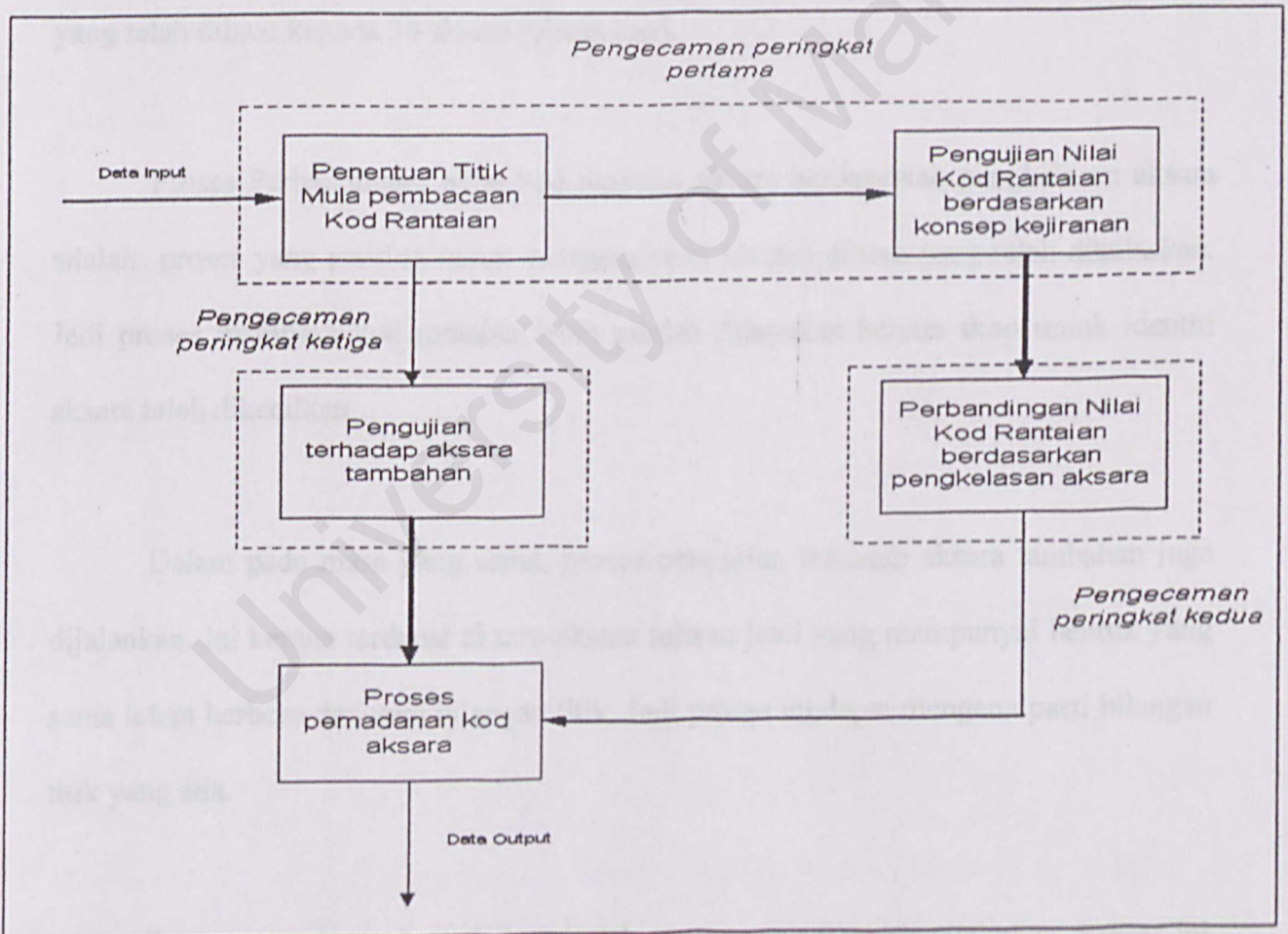
## 4.0 REKABENTUK SISTEM

Fasa Rekabentuk Sistem dilaksanakan untuk membangunkan suatu rekabentuk fizikal berasaskan rekabentuk logik sistem dengan menimbangkan keperluan yang telah ditetapkan dalam fasa analisis sistem.

Aktiviti-aktiviti rekabentuk sistem boleh dibahagikan kepada tiga bahagian :

- A. Rekabentuk gambaran keseluruhan perjalanan proses secara ringkas
- B. Rekabentuk bercirikan gambarajah input/ouput.
- C. Rekabentuk berasaskan Gambarajah blok.

### 4.1 Rekabentuk Gambaran Keseluruhan Perjalanan Proses Secara Ringkas



Rajah 12 : Gambarajah aliran perjalanan proses keseluruhan.

Rajah 12 menunjukkan gambaran perjalanan keseluruhan proses secara ringkas. Data akan diterima daripada proses peningkatan kualiti imej (Image Enhancement) iaitu proses yang dahulu dijalankan sebelum data imej menjalani proses pengecaman. Selepas itu, Proses penentuan titik mula dilakukan. Ini bertujuan untuk memudahkan pembacaan nilai piksel dengan langkah yang telah ditentukan dan mengelakkan ralat pembacaan

#### Proses Hough Transform

Selepas itu Proses pengujian nilai kod rantaian berdasarkan konsep kejiranan dilakukan. Proses ini adalah langkah utama dalam untuk mengenalpastian aksara-aksara berdasarkan ciri-ciri yang ada pada aksara tersebut. Ini boleh dirujuk pada pengkelasan yang telah dibuat kepada 36 aksara tulisan jawi.

Proses Perbandingan Nilai kod rantaian aksara berdasarkan pengkelasan aksara adalah proses yang penting untuk mengenalpasti identiti aksara yang telah dikelaskan. Jadi proses membaca kod rantaian lebih mudah dilakukan kerana skop untuk identiti aksara telah dikecilkan.

Dalam pada masa yang sama, proses pengujian terhadap aksara tambahan juga dijalankan. Ini kerana terdapat aksara-aksara tulisan jawi yang mempunyai bentuk yang sama tetapi berbeza dari segi bilangan titik. Jadi proses ini dapat mengenalpasti bilangan titik yang ada.

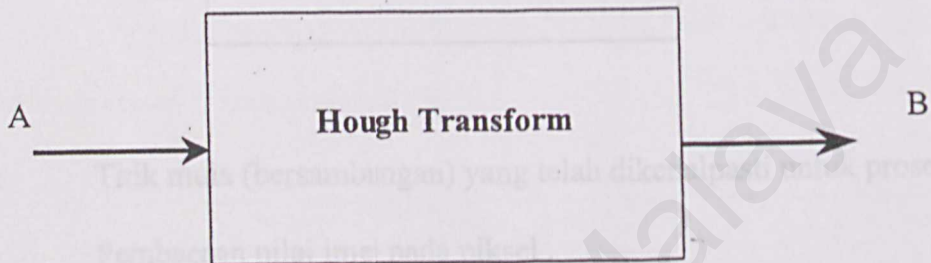
Proses pemadanan kod aksara adalah proses yang terakhir dilakukan. Proses ini bertujuan untuk mengenalpasti nilai kod bagi aksara tambahan dan kod aksara asas dan



gabungan nilai kod ini akan dirujuk pada pangkalan data untuk menghasilkan perwakilan kod yang tunggal.

## 4.2 Rekabentuk bercirikan gambarajah input/output

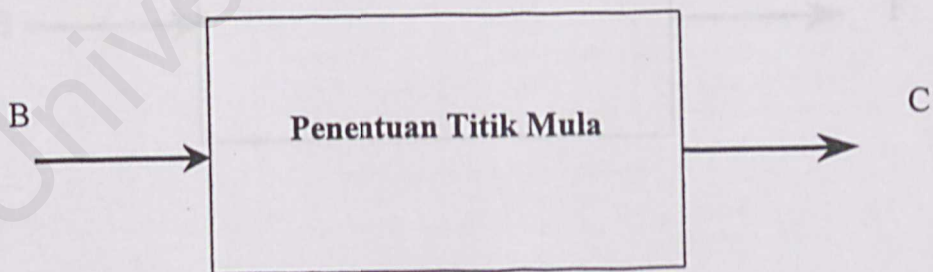
### Proses Hough Transform



Input A : Output bagi proses terdahulu iaitu Peningkatan kualiti imej (*Image Enhancement*)

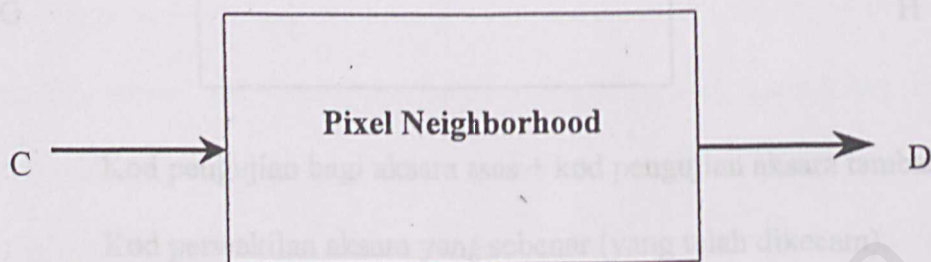
Output B : Data imej berada pada kedudukan yang dikehendaki (kedudukan yang betul).

### Proses Menentukan titik mula



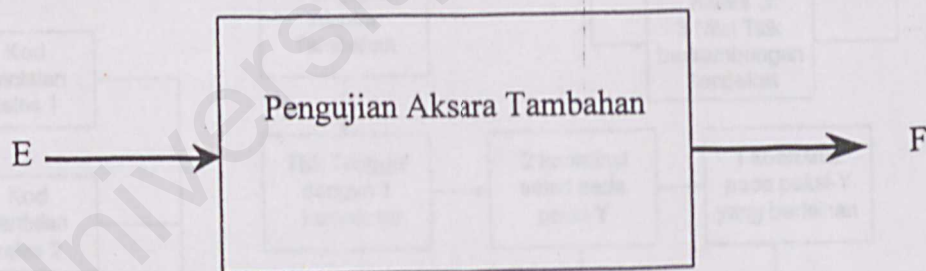
Input B : Output bagi proses Hough Transform

Output C : Titik mula dikenalpasti sebagai titik piksel teratas dan terkanan (bersambungan) atau sebarang titik tunggal (*dot*).

Proses PemadatanProses Berkaitan Kod Rantai

Input C : Titik mula (bersambungan) yang telah dikenalpasti untuk proses Pembacaan nilai imej pada piksel .

Output D : Keputusan algoritma pembacaan imej pada piksel (kod rantai).

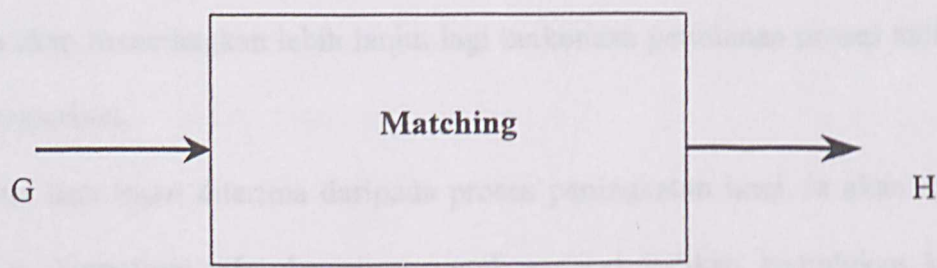
Pengujian Aksara Tambahan

Input E : Titik tunggal (dot) yang telah dikenalpasti .

Output F : Kod perwakilan hasil pengujian dijangka dalam bentuk perdua.

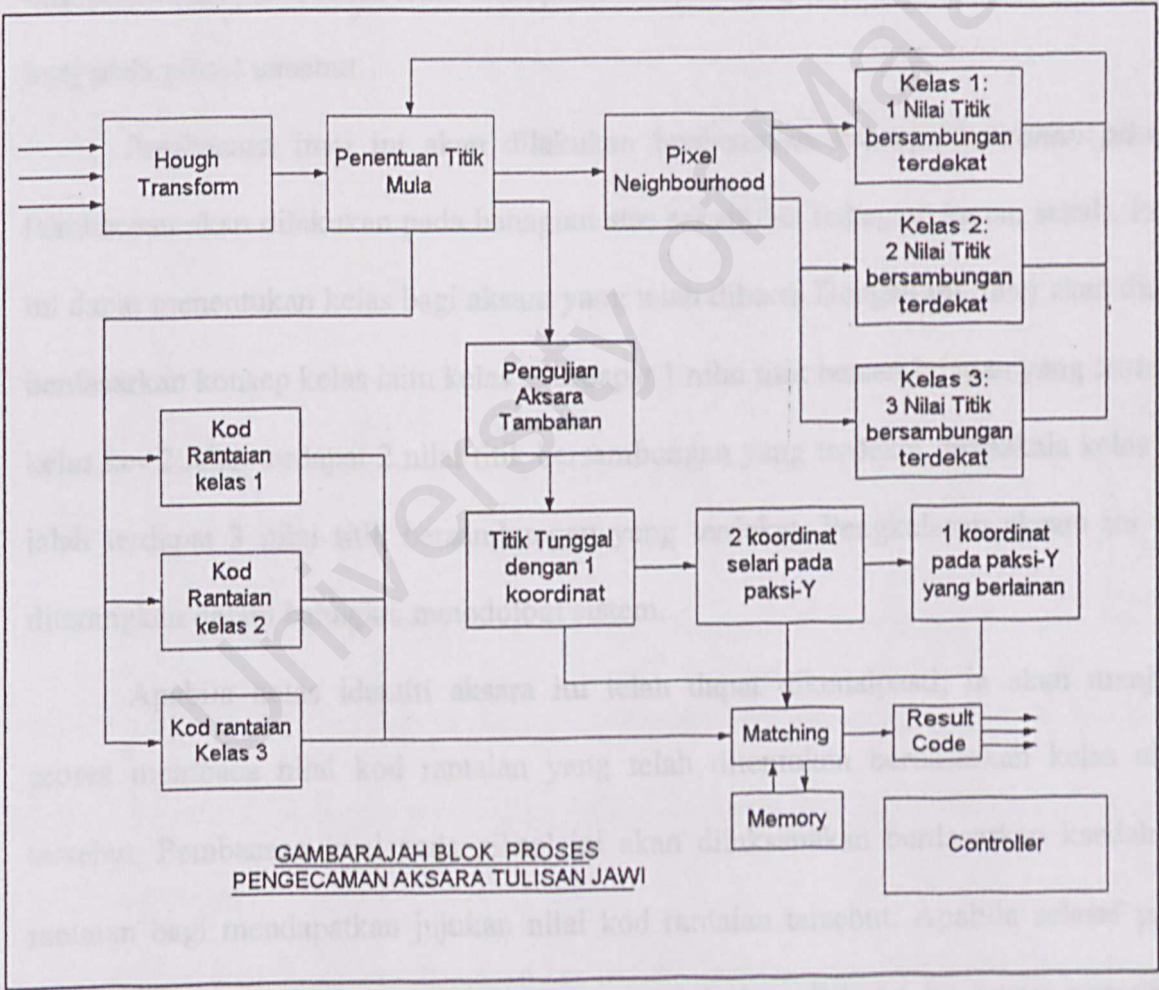


Proses Pemasanan



- Input G : Kod pengujian bagi aksara asas + kod pengujian aksara tambahan.
- Output H : Kod perwakilan aksara yang sebenar (yang telah dikecam).

4.3 Rekabentuk Berasaskan Gambarajah Blok



Rajah 13 : Gambarajah Blok Proses Pengesanan Aksara Tulisan Jawi.

Rajah 13 menunjukkan gambarajah blok proses pengecaman aksara tulisan jawi. Kali ini, saya akan menerangkan lebih lanjut lagi berkenaan perjalanan proses kajian ini secara lebih terperinci.

Apabila data input diterima daripada proses peningkatan imej, ia akan melalui proses 'Hough Transform'. Ini bertujuan untuk mengembalikan kedudukan kepada keadaan normal sekiranya imej yang diterima bukan pada keadaan yang tepat seperti senget atau terbalik. Selepas itu, data akan menjalani proses penentuan titik mula dimana pada proses ini, penetapan pembacaan imej telah ditetapkan. Jadi apabila kedudukan titik permulaan pembacaan telah ditetapkan ia akan mengelakkan ralat dalam pembacaan imej pada piksel tersebut.

Pembacaan imej ini akan dilakukan berdasarkan konsep '*kejiranan piksel*'. Pembacaan akan dilakukan pada bahagian atas sekali dan bahagian kanan sekali. Proses ini dapat menentukan kelas bagi aksara yang telah dibaca. Dengan ini, imej akan dirujuk berdasarkan konsep kelas iaitu kelas 1 terdapat 1 nilai titik bersambungan yang terdekat, kelas ke- 2 ialah terdapat 2 nilai titik bersambungan yang terdekat, manakala kelas ke-3 ialah terdapat 3 nilai titik bersambungan yang terdekat. Pengkelasan aksara ini telah diterangkan dalam bahagian metodologi sistem.

Apabila kelas identiti aksara itu telah dapat dikenalpasti, ia akan menjalani proses membaca nilai kod rangkaian yang telah ditentukan berdasarkan kelas aksara tersebut. Pembacaan imej pada piksel ini akan dilaksanakan berdasarkan kaedah kod rangkaian bagi mendapatkan jujukan nilai kod rangkaian tersebut. Apabila selesai proses membaca nilai imej pada piksel, hasil bacaan ini akan di bawa ke proses pemadanan aksara.



Semasa proses permulaan menentukan titik mula dan proses penentuan kelas aksara, proses pengujian terhadap aksara tambahan dilakukan. Ini kerana seperti yang kita telah sedia maklum bahawa aksara-aksara tulisan jawi yang unik ini mempunyai titik-titik dan ia dapat dibezakan melalui bilangan titik pada aksara tersebut. Proses ini akan mengesan nilai imej pada piksel. Merujuk kepada teori, aksara asas adalah jika nilai imej dibaca secara bersambungan, jadi sekiranya nilai imej itu dibaca tanpa secara bersambungan pada piksel itu, kita dapat menyatakan bahawa imej itu adalah aksara tambahan.

Bagi aksara tambahan., proses yang terlibat iaitu menentukan bilangan titik yang berkemungkinan ada 1 titik, 2 titik dan 3 titik. Pengujian ini telah diterangkan pada bahagian metodologi sistem. Selepas itu, hasil pembacaan titik ini dijangkakan adalah 3 bit dalam bentuk perduaan yang seterusnya akan dibawa ke proses pepadanan.

Proses pepadanan adalah merujuk kepada memori. Memori ini merupakan pangkalan data yang menyimpan keputusan bagi penggabungan nilai kod aksara asas dan aksara tambahan. Contohnya jika jujukan kod rangkaian (𐄇) ialah 6,6,5,4,4,3,2,2 dan digabungkan dengan kod aksara tambahan yang mempunyai 1 nilai titik akan dikenalpasti sebagai (𐄈). Output perwakilan aksara ini (Result Code) ini diwakili nombor. Contohnya huruf (𐄈) ini akan diwakili oleh nombor 2 tetapi input yang keluar adalah diwakili 2 sebagai mewakili (𐄈). Jadi output dapat dikatakan sebagai perwakilan aksara.

Diharapkan Rekabentuk sistem ini dapat diimplementasikan sebagaimana yang telah dirancang...*INSYAALLAH*...

Dalam bab ini, saya akan menerangkan kaedah satu algoritma yang digunakan dalam membangunkan proses pengiraan aksara tulisan jawi dengan menggunakan bahasa peritalan perkakasan "Very High Speed Integrated Circuits Hardware Description Language" (VHDL). Sebagaimana yang diketahui, dalam menjalankan proses pengiraan ini, saya telah menetapkan kaedah pembacaan melalui kaedah *Kod Rantian*. Jadi, matlamat output dalam proses pengiraan ini adalah hasil gabungan kod-kod rantian yang telah ditetapkan arah bagi setiap kod itu. Kod-kod itu pula akan dikeluarkan dalam perwakilan "Binary Code" iaitu bentuk 0 dan 1.

Saya telah menggunakan kaedah ini bagi mengimплеmentasikan kod-kod VHDL ini bagi menghasilkan output yang dikehendaki. Antaramuka yang digunakan adalah

## BAB 5

# PEMBANGUNAN SISTEM

- 1) Algoritma membaca input dan melakukan operasi aritmetik.
- 2) Algoritma membaca input dan melakukan operasi aritmetik.



Dalam bab ini, saya akan menerangkan kaedah atau algoritma yang digunakan dalam membangunkan proses pengecaman aksara tulisan jawi dengan menggunakan bahasa perihalan perkakasan “ Very High Speed Integrated Circuits Hardware Description Language” (VHDL). Sebagaimana yang diketahui, dalam menjalankan proses pengecaman ini, saya telah menetapkan kaedah pembacaan melalui kaedah **Kod Rantaian**. Jadi, matlamat output dalam proses pengecaman ini adalah hasil gabungan kod-kod rantaian yang telah ditetapkan arah bagi setiap kod itu. Kod-kod itu pula akan dikeluarkan dalam perwakilan ‘Binary Code ’ iaitu bentuk 0 dan 1.

Saya telah menggunakan beberapa pendekatan bagi mengimplementasikan kod-kod VHDL ini bagi menghasilkan output yang dikehendaki. Antara pendekatan yang digunakan adalah.

- 1) Algoritma membaca kod rantaian melalui Kaedah Tatasusunan.
- 2) Algoritma membaca kod rantaian melalui kejiranan piksel yang telah ditetapkan.

5.1

ALGORITMA MEMBACA KOD RANTAIAN MELALUI KAEDAH TATASUSUNAN

Kaedah pembacaan kod rantaian berdasarkan tatasusunan ini adalah dengan menakrifkan pembacaan nilai input binari yang diterima secara bacaan matrik. Setiap piksel akan diwakili oleh tatasusunan dua dimensi seperti rajah dibawah :

Penentuan Titik Mula

C(0,0)	C(0,1)	C(0,2)	C(0,3)
C(1,0)	C(1,1)	C(1,2)	C(1,3)
C(2,0)	C(2,1)	C(2,2)	C(2,3)
C(3,0)	C(3,1)	C(3,2)	C(3,3)

Rajah 5.1.1 : Cara pembacaan piksel bagi tatasunan 4 x 4



Rajah 5.1.1 menunjukkan bagaimana cara pembacaan piksel yang mewakili nilai input yang diterima secara berjujukan. Setiap piksel diwakili dengan 2 dimensi tatasusunan.

Piksel C(1,3) akan menjadi tempat di mana bermulanya semua proses pembacaan nilai input yang diterima. Setiap kemasukan 16 bit input akan dikira secara piksel ke piksel.

Sebagai contoh, jika input yang diterima sebanyak 16 bit iaitu 1001100110010110. Ia akan dibaca secara berturutan yang diwakili oleh piksel. Jika piksel = '1' ia mempunyai imej tetapi jika piksel itu = '0', ia tiada imej.

1	0	0	1
1	0	0	1
1	0	0	1
0	1	1	0

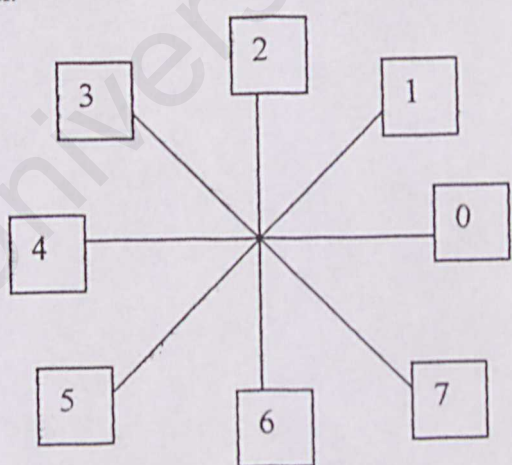
Penentuan  
Titik Mula



Rajah 5.1.2 : Contoh Nilai Binari 16 bit Input yang diterima

$C(0,0) = '1'$	$C(0,1) = '0'$
$C(0,2) = '0'$	$C(0,3) = '1'$
$C(1,0) = '1'$	$C(1,1) = '0'$
$C(1,2) = '0'$	$C(1,3) = '1'$
$C(2,0) = '1'$	$C(2,1) = '0'$
$C(2,2) = '0'$	$C(2,3) = '1'$
$C(3,0) = '1'$	$C(3,1) = '1'$
$C(3,2) = '1'$	$C(3,3) = '1'$

Jadi apabila input diterima, ia akan dibaca pada piksel yang diwakili  $C(0,3)$ . Dalam proses pembacaan nilai pada piksel ini, saya telah menetapkan titik  $C(0,3)$  sebagai titik permulaan yang akan membaca setiap piksel tersebut. Penentuan titik mula aksara asas mestilah dibaca dari arah kanan ke kiri berdasarkan tulisan jawi ditulis dan dibaca. Proses pembacaan setiap piksel ini akan melalui syarat-syarat yang telah ditetapkan bagi menakrifkan nilai kod rantaian yang wujud. Seperti yang diketahui kod rantaian ini mempunyai lapan arah.



Rajah 5.1.3 : Gambarajah arah Kod Rantaian



```
library ieee;
```

```
use ieee.std_logic_1164.all;
```

```
-- use ieee.numeric_std.all;    -- Note: uncomment this if you use
```

```
    -- IEEE standard signed or unsigned types.
```

```
-- use ieee.std_logic_arith.all; -- Note: uncomment this if you use
```

```
    -- Synopsys signed or unsigned types.
```

```
entity TITIKMULA is
```

```
    port (
```

```
        -- WIZ BEGINPORTS (Entity Wizard command)
```

```
        a: in std_logic;
```

```
        clk: in std_logic;
```

```
        reset: in std_logic    ;
```

```
        value: out std_logic_vector(2 downto 0)
```

```
        -- WIZ ENDPORTS (Entity Wizard command)
```

```
    );
```

```
end TITIKMULA;
```

```

library ieee;

use ieee.std_logic_1164.all;

-- use ieee.numeric_std.all; -- Note: uncomment this if you use
-- IEEE standard signed or unsigned types.

-- use ieee.std_logic_arith.all; -- Note: uncomment this if you use
-- Synopsys signed or unsigned types.

entity TITIKMULA is
port (
-- WIZ BEGINPORTS (Entity Wizard command)

a: in std_logic;

clk: in std_logic;

reset: in std_logic ;

value: out std_logic_vector(2 downto 0)

-- WIZ ENDPORTS (Entity Wizard command)

);

end TITIKMULA;

```



**Pengisytiharan matric a 4 x 4**

“type matric\_a is array(0 to 15,15 downto 0) of bit;

signal c:matric\_a;”

Penerangan : matric\_a adalah merangkap kepada nilai input a yang diterima dan diwakili 2 dimensi tatasusunan. Sebagaimana dalam aturcara C atau Mathlab yang mempunyai baris dan lajur, jadi (0 to 15,15 downto 0) mewakili baris dan lajur bagi matrik 4 x 4. Baris (0 to 15) akan dibaca dari atas ke bawah manakala lajur (15 downto 0) akan dibaca dari kanan ke kiri kerana protokol tulisan jawi dibaca dari kanan ke kiri. Manakala signal C pula berfungsi sebagai pembawa nilai matric\_a, ‘ C(0 to 15,15downto 0) ’.

**Pengecaman Kod Rantaian**

“if (reset = '1')then

value<="000";

elsif (clk'event AND clk= '1') then

if (c(0,3)='1' and c(0,2)='1') then

value<="100";

elsif (c(0,3)='1' and c(1,2)='1')then

value<="101";

elsif (c(0,3)='1' and c(1,3)='1')then

value<="110";

end if;

Penerangan : Bahagian ini adalah merujuk kepada titik mula dan pengecaman kod rantaian berdasarkan kejiranan piksel. Bagi kes di atas :

Apabila  $c(0,3)$  bernilai dan  $c(0,2)$  juga bernilai, arah bacaan adalah  $\leftarrow$ . Jadi nilai kod rantaian adalah 4 dan merujuk kepada value = "100".

Apabila  $c(0,3)$  bernilai dan  $c(1,2)$  juga bernilai, arah bacaan adalah  $\swarrow$ . Jadi nilai kod rantaian adalah 5 dan merujuk kepada value = "101".

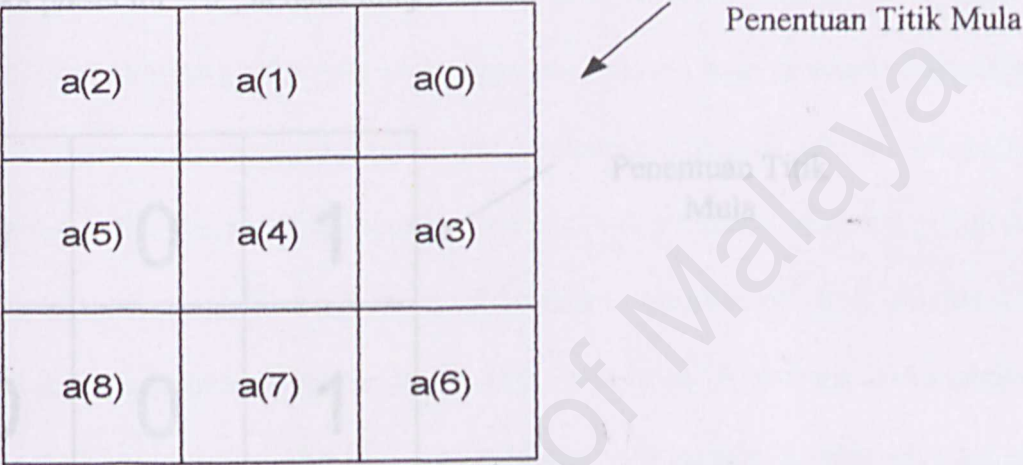
Apabila  $c(0,3)$  bernilai dan  $c(1,3)$  juga bernilai, arah bacaan adalah  $\downarrow$ . Jadi nilai kod rantaian adalah 6 dan merujuk kepada value = "110".

Jadi melalui algoritma ini, output akan keluar secara jujukan kod rantaian tetapi ini adalah dalam proses algoritma dan keputusan sebenar akan diketahui dalam bab Pengujian Sistem nanti.



5.2 ALGORITMA MEMBACA KOD RANTAIAN BERDASARKAN KEJIRANAN PIKSEL YANG TELAH DITETAPKAN

Kaedah pembacaan kod rantain berdasarkan kejiranan piksel ini adalah dengan menakrifkan pembacaan nilai input binari yang diterima secara piksel ke piksel. Setiap piksel akan membaca setiap kemasukan input binari yang telah ditetapkan seperti rajah dibawah :

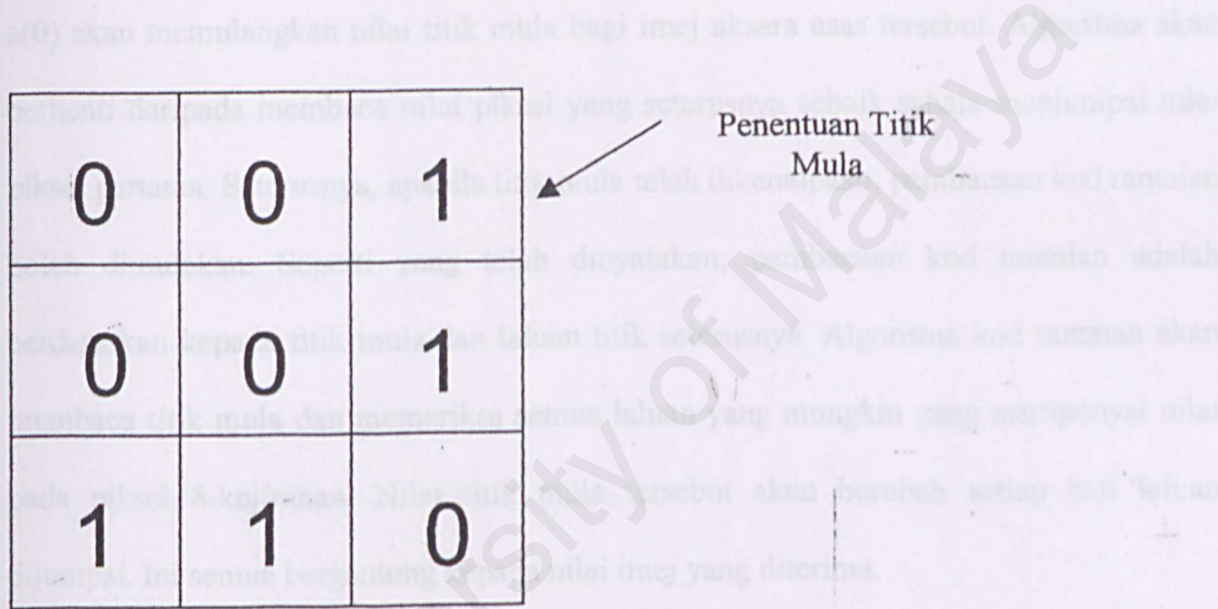


Rajah 5.2.1 : Cara pembacaan bagi setiap piksel ke piksel berdasarkan konsep kejiranan 3 x 3

Rajah 5.2.1 menunjukkan bagaimana cara pembacaan piksel yang mewakili nilai input yang diterima secara berjujukan. Setiap piksel telah diwakili oleh kemasukan setiap input.

Proses pembacaan ini adalah hampir sama dengan algoritma tatasusunan tadi iaitu terdapat satu titik yang akan menjadi titik permulaan pembacaan. Pada algoritma ini, Pikel a(0) akan menjadi tempat dimana bermulanya semua proses pembacaan nilai input yang diterima. Setiap kemasukan 9 bit input akan dikira secara piksel ke piksel.

Sebagai contoh, jika input yang diterima sebanyak 9 bit iaitu 100100011. Ia akan dibaca secara berturutan yang diwakili oleh piksel. Jika piksel = '1' ia mempunyai imej tetapi jika piksel itu = '0', ia tiada imej.



Rajah 5.2.2 : Contoh Nilai Binari 9 bit Input yang diterima

- a(0) = '1'
- a(1) = '0'
- a(2) = '0'
- a(3) = '1'
- a(4) = '0'
- a(5) = '0'
- a(6) = '0'
- a(7) = '1'
- a(8) = '1'



Apabila input diterima, ia akan dibaca pada piksel yang diwakili  $a(0)$ . Dalam proses pembacaan nilai pada piksel ini, saya telah menetapkan titik  $a(0)$  sebagai titik permulaan yang akan membaca setiap piksel tersebut. Penentuan titik mula aksara asas mestilah dibaca dari arah kanan ke kiri berdasarkan tulisan jawi ditulis dan dibaca. Apabila Proses pembacaan setiap piksel ini akan melalui syarat-syarat yang telah ditetapkan bagi menakrifkan nilai kod rangkaian yang wujud.

Dalam algoritma titik mula ini, nilai pertama yang dijumpai pada piksel  $a(0)$  akan memulangkan nilai titik mula bagi imej aksara asas tersebut. Algoritma akan berhenti daripada membaca nilai piksel yang seterusnya sebaik sahaja menjumpai nilai piksel pertama. Seterusnya, apabila titik mula telah dikenalpasti, pembacaan kod rangkaian boleh dimulakan. Seperti yang telah dinyatakan, pembacaan kod rangkaian adalah berdasarkan kepada titik mula dan laluan titik seterusnya. Algoritma kod rangkaian akan membaca titik mula dan memeriksa semua laluan yang mungkin yang mempunyai nilai pada piksel 8-kejiranan. Nilai titik mula tersebut akan berubah setiap kali laluan dijumpai. Ini semua bergantung kepada nilai imej yang diterima.

### 5.2.3 pengaturcaraan bagi pengecaman kod Rangkaian

---

```
-- Auto-generated module template: CHAIN
-- chain1 = "001"
-- Note: comments beginning with WIZ should be left intact.
-- Other comments are informational only.
-- PeakVHDL software version: 5.20c
```

```

library ieee;

use ieee.std_logic_1164.all;

-- use ieee.numeric_std.all;    -- Note: uncomment this if you use
                                -- IEEE standard signed or unsigned types.

-- use ieee.std_logic_arith.all; -- Note: uncomment this if you use
                                -- Synopsys signed or unsigned types.

```

entity CHAIN is

```

port (
    -- WIZ BEGINPORTS (Entity Wizard command)

    a: in std_logic_vector(8 downto 0);

    clk: in std_logic;

    reset: in std_logic;

    value: out std_logic_vector(2 downto 0)

    -- WIZ ENDPORTS (Entity Wizard command)

);

```

end CHAIN;

architecture BEHAVIOR of CHAIN is

```

-- Note: signals, components and other objects may be declared here if needed.

--chain1:="001";

--chain2:="010";

--chain3:="011";

--chain4:="100";

--chain5:="101";

```



```

--chain6:="110";
--chain7:="111";

begin

process(clk,reset,a(8 downto 0))

begin

if (reset = '1')then

    value<="000";

elseif rising_edge(clk)then

    if (a(0)='1' and a(1)='1') then -- assign a(0) for the first point and define the chain

        value<="100";

    --case (a(1) and a(2))='1'

        if (a(1)='1' and a(2)='1')then

            value<="100";

            if (a(2)='1' and a(4)='1')then

                value<="111";

                if(a(4)='1' and a(6)='1')then

                    value<="111";

                    if (a(4)='1' and a(7)='1')then

                        value<="110";

                        if (a(2)='1' and a(5)='1')then

                            value<="110";

                            if (a(5)='1' and a(8)='1')then

                                value<="110";

```

```
if (a(5)='1' and a(7)='1')then
```

```
value<="111";
```

```
if (a(1)='1' and a(4)='1')then
```

```
value<="110";
```

```
end if;
```

```
end if;
```

```
--case (a(1) and
```

```
if (a(1)='1' and a(4)='1')then
```

```
value<="110";
```

```
if (a(4)='1' and
```

```
value<="100";
```

```
end if;
```

```
--case (a(1) and a(3))='1'
```

```
if (a(1)='1' and a(3)='1')then
```

```
value<="111";
```

```
if (a(3)='1' and a(6)='1')then
```

```
value<="110";
```

```
if(a(6)='1' and a(7)='1')then
```

```
end if;
```

```
--case (a(1) and a(3))='1'
```

```
if (a(1)='1' and a(5)='1')then
```

```
value<="101";
```



```

    if (a(7)='1' and a(8)='1')then
        value<="100";
    end if;
end if;

--case (a(1) and a(4)='1'
if (a(1)='1' and a(4)='1') then
    value<="110";

    if (a(4)='1' and a(5)='1')then
        value<="100";
    end if;

    if (a(4)='1' and a(7)='1')then
        value<="110";
    end if;

    if (a(4)='1' and a(8)='1')then
        value<="101";
    end if;
end if;

end if;

--case (a(1) and a(5)='1'-
if (a(1)='1' and a(5)='1')then
    value<="101";
end if;

```

if (a(5)='1' and a(8)='1')then

value<="110";

if (a(5)='1' and a(7)='1')then

value<="111";

end if;

end if;

end if;

end if;

end if;

if (a(0)='1' and a(3)='1') then -- assign a(0) for the first point and define the

value<="110"; chain code to a(3)

--case (a(3) and a(4))='1'

if(a(3)='1' and a(4)='1')then

value<="100";

if (a(4)='1' and a(5)='1')then

value<="100";

if (a(5)='1' and a(7)='1')then

value<="111";

if(a(5)='1' and a(8)='1')then

value<="110";

if (a(4)='1' and a(7)='1')then

value<="110";





```
if(a(7)='1' and a(8)='1')then
    value<="100";
end if;

    end if;

end if;

--case (a(4) and a(5))='1'
end if;

--case a(3)='1' and a(7)='1'
if (a(3)='1' and a(7)='1')then
    value<="101";
    if (a(7)='1' and a(8)='1')then
        value<="100";
    end if;
end if;

if (a(0)='1' and a(4)='1') then -- assign a(0) for the first point and define
    value<="101";           the chain code to a(4)
--case (a(4) and a(5))='1'
if (a(4)='1' and a(5)='1')then
    value<="100";
    if (a(5)='1' and a(7)='1')then
        value<="111";
    end if;
end if;
```



```

    if(a(5)='1' and a(8)='1')then
        value<="110";
    end if;
end if;
end if;
end if;

--case (a(4) and a(7)='1'
if (a(4)='1' and a(7)='1')then
    value<="110";
    if (a(7)='1' and a(8)='1')then
        value<="100";
    end if;
end if;

--case (a(4) and a(6)='1'
if (a(4)='1' and a(6)='1')then
    value<="111";
    if (a(6)='1' and a(7)='1')then
        value<="100";
    if (a(7)='1' and a(8)='1')then
        value<="100";
    end if;
    end if;
end if;
end if;
```

```

    if (a(4)='1' and a(8)='1') then
        value<="101";
    end if;

end if;

end if;

-- Note: concurrent statements (including concurrent assignments
-- and component instantiations) go here.

end process;

end BEHAVIOR;

```

Aturcara di atas ini adalah draf ringkas gambaran perjalanan input. Saya akan menerangkan sinopsis perjalanan proses bagi draf aturcara di atas. Terdapat 3 input dan 1 output iaitu:

a : input yang diterima  
 clk : input bagi Clock Per Cycle  
 Reset : Input bagi reset sistem  
 Value : Kod Rantaian yang dikenalpasti

“ process (clk,reset,a(8 downto 0)) ”

Penerangan : Dibahagian proses ini adalah perwakilan nilai a sebanyak 9 bit yang telah ditetapkan bagi pembacaan setiap input imej pada piksel.

**Pengecaman Kod Rantaian**

if (a(0)='1' and a(1)='1') then -- assign a(0) for the first point and define the chain

value<="100";                      code to a(1)

--case (a(1) and a(5)='1'-

if (a(1)='1' and a(5)='1')then

value<="101";

if (a(5)='1' and a(8)='1')then

value<="110";

if (a(5)='1' and a(7)='1')then

value<="111";

end if;

end if;

end if;

Penerangan : Aturcara di atas adalah syarat pembacaan nilai kod rantaian berdasarkan kejiranan piksel yang bernilai. Bagi pernyataan bersyarat di atas, kaedah pembacaannya seperti berikut:

Apabila a(0) bernilai dan a(1) juga bernilai, arah bacaan adalah ← . Jadi nilai kod rantaian adalah 5 dan merujuk kepada value = "100".

Disini, oleh kerana a(1) bernilai maka ia akan menjadi titik mula yang baru dan akan memeriksa kejiranan piksel yang bernilai( wujudnya imej) bagi mengecam nilai kod rantaian.



Apabila  $a(1)$  bernilai dan  $a(5)$  juga bernilai, arah bacaan adalah.  
kod rantaiian adalah 5 dan merujuk kepada value = "101".



Jadi nilai

Apabila  $a(5)$  bernilai dan  $a(8)$  juga bernilai, arah bacaan adalah.  
nilai kod rantaiian adalah 6 dan merujuk kepada value = "110".



Jadi

Apabila  $a(5)$  bernilai dan  $a(7)$  juga bernilai, arah bacaan adalah.  
nilai kod rantaiian adalah 7 dan merujuk kepada value = "111".



Jadi

Sebagai contoh input, jika  $a(0) = '1'$ ,  $a(1) = '1'$ ,  $a(5) = '1'$  dan  $a(8) = '1'$  maka gambaran input yang diterima adalah seperti berikut :

0	1	1
1	0	0
1	0	0

Pembacaan piksel  
bermula disini



Jadi output yang diterima sebagai kod rantaiian dalam perwakilan binari adalah '100', '101', '110'. Kesimpulannya , algoritma ini bermatlamatkan output yang terhasil adalah dalam bentuk gabungan kod rantaiian dalam perwakilan binari yang mewakili imej pada piksel. Dalam bab Pengujian Sistem kita akan dapat mengetahui simulasi algoritma ini.

Dalam Bab Pengujian Sistem ini saya akan menerangkan proses yang telah dilaksanakan pada algoritma-algoritma tadi dan keputusan pengujannya. Seperti yang telah sedia maklum, saya telah menggunakan bahasa perihalan perkakasan VHDL bagi medium pengantara setiap algoritma. Jadi 'Compiler' yang telah saya gunakan adalah perisian *PEAKFPGA* yang popular digunakan pada masa kini bagi pengkompilan bahasa perihalan perkakasan.

## 6.1 PENGUJIAN ALGORITMA MEMBACA KOD RANTAIAN MELALUI KAEDAH TATASUSUNAN

Di bawah ini adalah antara lain yang akan dijalankan juga kesudahan simulasi bagi program algoritma membaca kod rantai melalui kaedah tatasusunan. Saya hanya menguji pembacaan tajuk pada piksel yang awal sahaja.

### 6.1.1 PENGUJIAN SISTEM

-- Auto-generated test values are from the JIS-TITIKMULA

-- Note: comments values are with WIZ should values left intact.

-- Other comments are informational only.

-- Peak VHDL software version: 2.20c

Dalam Bab Pengujian Sistem ini saya akan menerangkan proses yang telah dilaksanakan pada algoritma-algoritma tadi dan keputusan pengujiannya. Seperti yang telah sedia maklum, saya telah menggunakan bahasa perihalan perkakasan VHDL bagi medium pengantara setiap algoritma. Jadi 'Compiler' yang telah saya gunakan adalah perisian **PEAKFPGA** yang popular digunakan pada masa kini bagi pengkompilan bahasa perihalan perkakasan.

## 6.1 PENGUJIAN ALGORITMA MEMBACA KOD RANTAIAN MELALUI KAEDAH TATASUSUNAN

Di bawah ini adalah aturcara bagi pengujian dan juga keputusan simulasi bagi program algoritma membaca kod rantaiian melalui kaedah tatasusunan. Saya hanya menguji pembacaan imej pada piksel yang awal sahaja dahulu.

### 6.1.1 Aturcara Pengujian

```
-----
-- Auto-generated test valueench template for: TITIKMULA
--
-- Note: comments valueeginning with WIZ should valuee left intact.
-- Other comments are informational only.
--
-- PeakVHDL software version: 5.20c
--
```



```
library ieee;

use ieee.std_logic_1164.all;

library ieee;

use ieee.std_logic_1164.all;

-- use ieee.numeric_std.all; -- Note: uncomment this if you use
-- IEEE standard signed or unsigned types.

-- use ieee.std_logic_arith.all; -- Note: uncomment/modify this if you use
-- Synopsys signed or unsigned types.

use std.textio.all;

use work.TITIKMULA;

entity TESTBENCH is

end TESTBENCH;

architecture stimulus of TESTBENCH is

component TITIKMULA is

    port (

        a: in std_logic;

        value: out std_logic;

        clk: in std_logic;

        reset: in std_logic

    );

end component;
```

```
constant PERIOD: time := 70 ns;
```

```
-- Top level signals go here...
```

```
signal a: std_logic;
```

```
signal value: std_logic_vector(2 downto 0);
```

```
signal clk: std_logic;
```

```
signal reset: std_logic;
```

```
signal done: boolean := false;
```

```
signal s_clock: natural := 0;
```

```
begin
```

```
    DUT: TITIKMULA port map (
```

```
        a,
```

```
        value,
```

```
        clk,
```

```
        reset
```

```
    );
```

```
CLOCK1: process
```

```
begin
```

```
    s_clock <= s_clock + 1;
```

```
    clk <= '1';
```

```
    wait for 25 ns;
```

```
    clk <= '0';
```

```
    wait for 25 ns;
```

```
end process CLOCK1;
```

STIMULUS1: process

begin

reset<='1'; -- Reset the system

wait for PERIOD; -- Wait one clock cycle

reset <= '0'; -- de-assert reset

a <= '1';

wait for PERIOD;

a <= '0';

wait for PERIOD;

a <= '1';

wait for PERIOD;

a <= '1';

wait for PERIOD;

done <= true; -- Turn off the clock

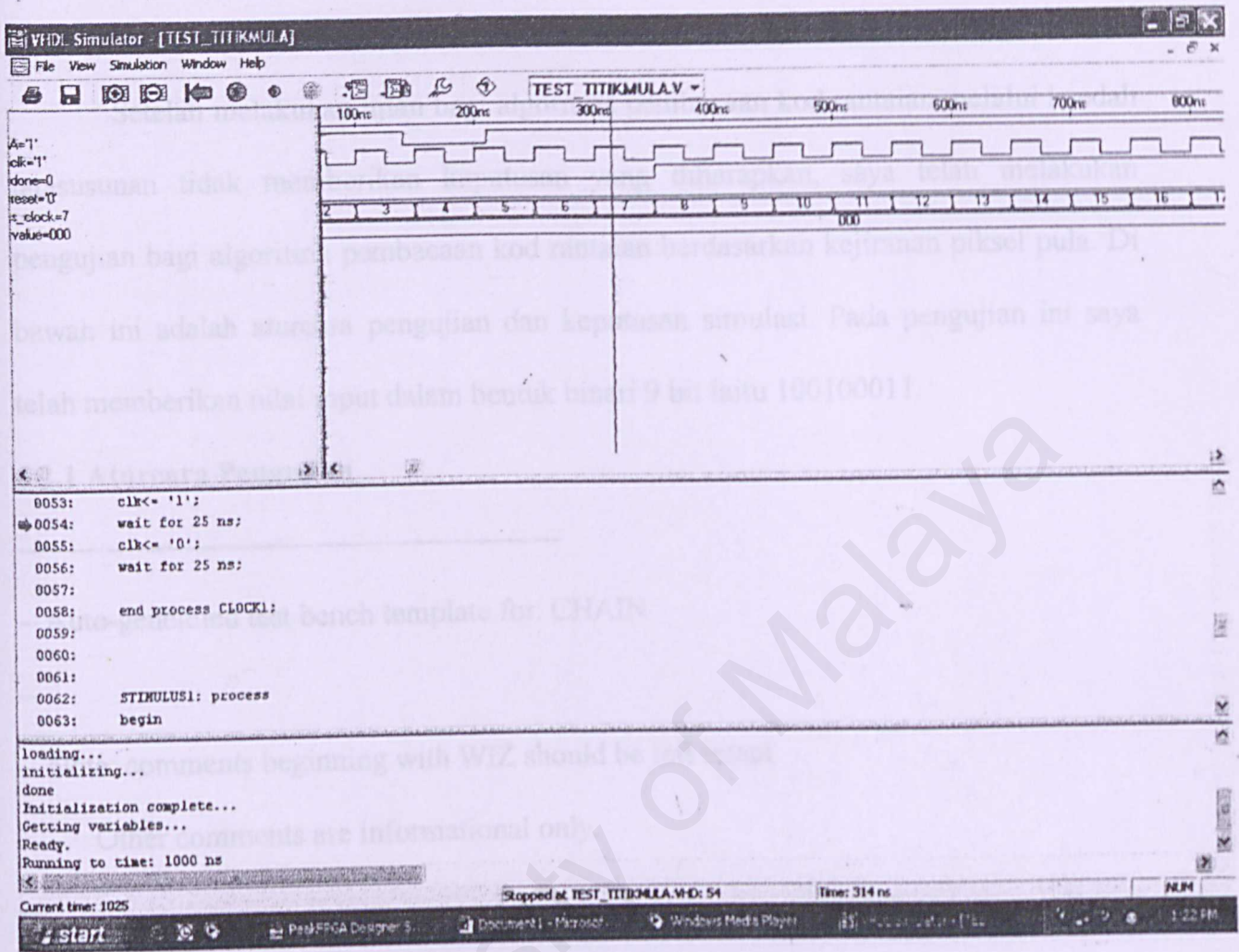
wait; -- Suspend simulation

end process STIMULUS1;

end stimulus;



6.1.2 Keputusan Simulasi



Rajah 6.1.3 Rajah bagi simulasi

6.1.4 Kesimpulan Simulasi

Setelah membuat pengujian pada algoritma dengan menggunakan *Testbench* dan juga membuat proses simulasi, keputusan tidak dapat mematuhi teori awal algoritma yang telah saya rancang pada bab pembangunan sistem. Output yang terhasil tidak mengeluarkan jujukan kod rantai seperti yang diharapkan. Oleh kerana pada peringkat awal lagi imej tidak dapat dibaca, saya membuat keputusan menggunakan algoritma yang lain. Jadi kesilapan pada algoritma akan dibincangkan pada bab Perbincangan Sistem.

## 6.2 PENGUJIAN ALGORITMA PEMBACAAN KOD RANTAIAN BERDASARKAN KEJIRANAN PIKSEL

Setelah melakukan ujian bagi algoritma pembacaan kod rantaiian melalui kaedah tatasusunan tidak memberikan keputusan yang diharapkan, saya telah melakukan pengujian bagi algoritma pembacaan kod rantaiian berdasarkan kejiranan piksel pula. Di bawah ini adalah aturcara pengujian dan keputusan simulasi. Pada pengujian ini saya telah memberikan nilai input dalam bentuk binari 9 bit iaitu 100100011.

### 6.2.1 Aturcara Pengujian

```

-----
-- Auto-generated test bench template for: CHAIN
--
-- Note: comments beginning with WIZ should be left intact.
-- Other comments are informational only.
-- Top level signals go here...
-- PeakVHDL software version: 5.20c
--
library ieee;
use ieee.std_logic_1164.all;
-- use ieee.numeric_std.all; -- Note: uncomment this if you use
-- IEEE standard signed or unsigned types.
-- use ieee.std_logic_arith.all; -- Note: uncomment/modify this if you use
-- Synopsys signed or unsigned types.
use std.textio.all;

```



use work.CHAIN;

entity TESTBNCH is

end TESTBNCH;

architecture stimulus of TESTBNCH is

component CHAIN is

port (

a: in std\_logic\_vector(8 downto 0);

clk: in std\_logic;

reset: in std\_logic;

value: out std\_logic\_vector(2 downto 0)

);

end component;

constant PERIOD: time := 100 ns;

-- Top level signals go here...

signal a: std\_logic\_vector(8 downto 0);

signal clk: std\_logic;

signal reset: std\_logic;

signal value: std\_logic\_vector(2 downto 0);

signal pixel:natural:=0;

signal done: boolean := false;



begin

DUT: CHAIN port map (

a,

clk,

reset,

value

);

CLOCK1: process

begin

pixel<= pixel + 1;

clk<= '1';

wait for 50 ns;

clk<= '0';

wait for 50 ns;

end process CLOCK1;

STIMULUS1: process

Begin

-- reset<='1'; -- Reset the system

-- a <="";

-- wait for PERIOD; -- Wait one clock cycle

--reset<='1'; -- reset <= '0'; -- de-assert reset

reset<='1';

wait for PERIOD;

```
reset<='0';
```

```
a(0)<='1';
```

```
wait for PERIOD;
```

```
a(1)<='0';
```

```
wait for PERIOD;
```

```
a(2)<='0';
```

```
wait for PERIOD;
```

```
a(3)<='1';
```

```
wait for PERIOD;
```

```
a(4)<='0';
```

```
wait for PERIOD;
```

```
a(5)<='0';
```

```
wait for PERIOD;
```

```
a(6)<='0';
```

```
wait for PERIOD;
```

```
a(7)<='1';
```

```
wait for PERIOD;
```

```
a(8)<='1';
```

```
wait for PERIOD;
```

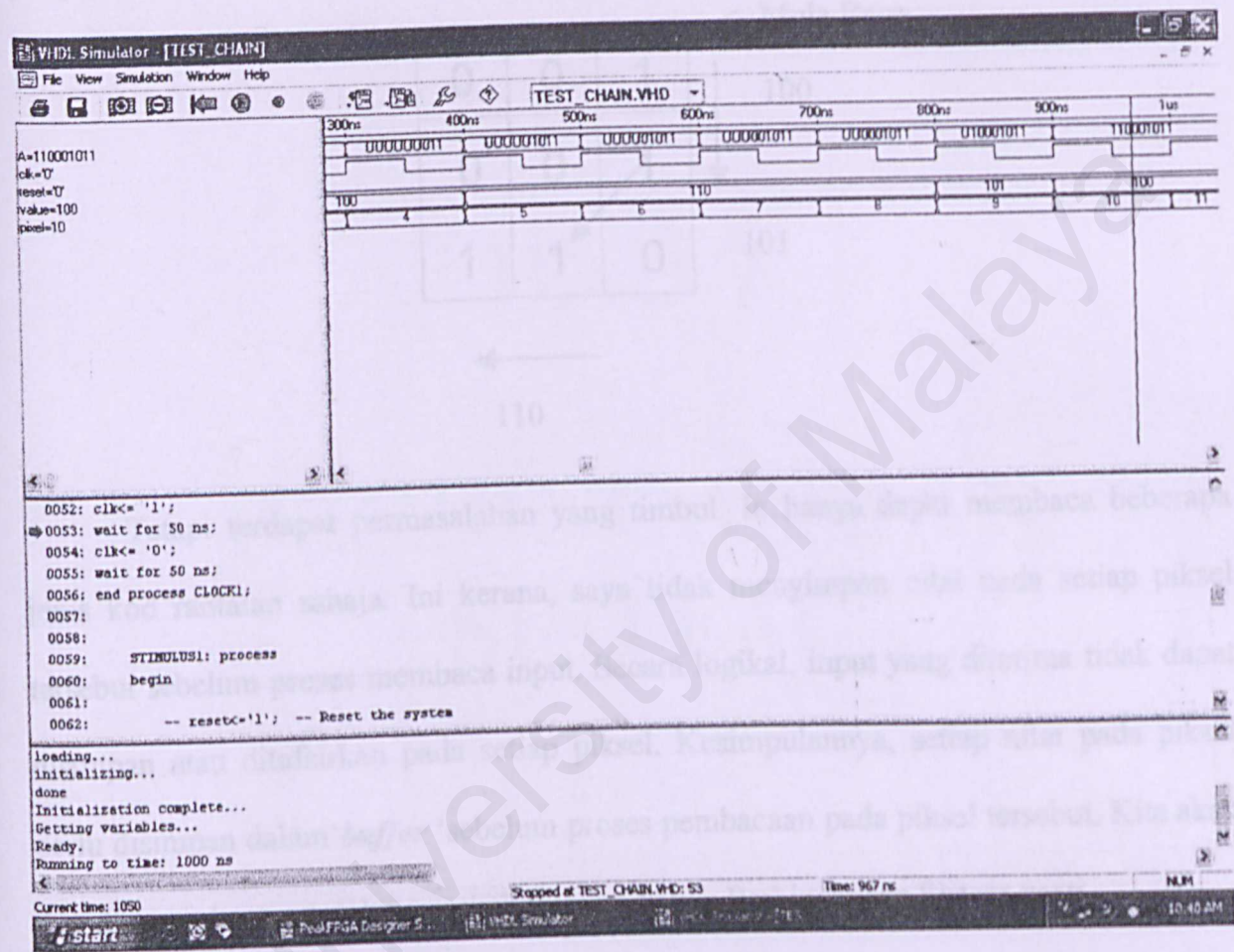
```
done <= true; -- Turn off the clock

wait; -- Suspend simulation

end process STIMULUS1;

end stimulus;
```

6.2.2 Keputusan Simulasi



Rajah 6.2.3 Rajah bagi simulasi

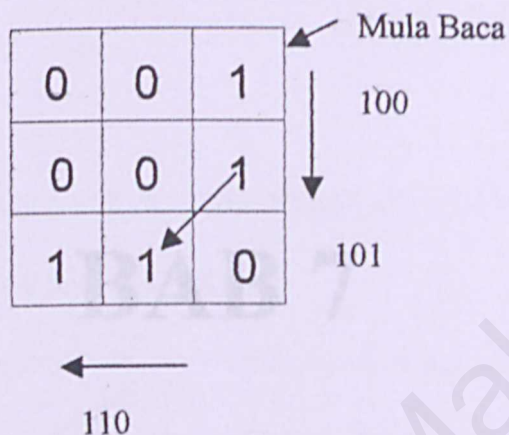
6.2.4 Kesimpulan Bagi Simulasi

Setelah melakukan simulasi pada algoritma ini, output dapat menghasilkan gabungan kod rantai iaitu seperti yang telah dirancang pada bab Pembangunan Sistem.



Rajah 6.2.3 menunjukkan bagaimana setiap input akan masuk ke setiap piksel demi piksel dan jenis kod rantaiannya akan diperiksa.

Pembacaan kod rantaiannya adalah berdasarkan nilai input yang diterima dan output yang dihasilkan adalah "100", "101" dan "110" merujuk kepada kod rantaiannya 4, 5 dan 6.



Tetapi terdapat permasalahan yang timbul. Ia hanya dapat membaca beberapa jenis kod rantaiannya sahaja. Ini kerana, saya tidak menyimpan nilai pada setiap piksel tersebut sebelum proses membaca input. Secara logikal, input yang diterima tidak dapat disimpan atau ditafsirkan pada setiap piksel. Kesimpulannya, setiap nilai pada piksel perlu disimpan dalam '*buffer*' sebelum proses pembacaan pada piksel tersebut. Kita akan membincangkan masalah yang timbul ini pada bab Perbincangan Sistem nanti.

Dalam Bab ini, saya akan membicarakan kelebihan, kelemahan dan masalah algoritma-algoritma yang timbul ketika melaksanakan proses dan simulasi bagi proses pengucapan aksara bahasa jawi dengan menggunakan bahasa perihalan perkakasan VHDL. Bab ini juga akan menyertakan cadangan bagi membentak proses atau algoritma yang lebih mantap dan boleh menghasilkan output yang sebarang ralat dan permasalahan.

## 7.1 PERBINCANGAN BAGI ALGORITMA PEMBACAAN KOD RANTAIAN

# BAB 7

MELALUI KEDAH TATA

## PERBINCANGAN SISTEM

Seperi yang telah kita maklum, proses pengucapan aksara bahasa jawi ini dengan mengimplementasikan kaedah tatasusunan telah dilaksanakan dalam bahasa pengaturcaraan VHDL. Namun begitu, dalam bab ini adalah berisikan pengaturcaraan perihalan. Namun begitu, dalam bab ini adalah berisikan pengaturcaraan perkakasan. Seperti yang telah kita maklum, kaedah tatasusunan dalam bahasa perihalan perkakasan VHDL bagi proses pembacaan kod rantai. Secara logiknya, kaedah ini boleh menghasilkan output yang diharapkan tanpa sebarang ralat dan rintang. Setelah itu, terdapat masalah yang timbul iaitu kaedah tatasusunan ini sendiri. Seperti yang telah kita maklum, dalam kaedah simulasi, output tidak dapat menghasilkan gabungan kod rantai. Jadi saya akan menerangkan bahagian dimana kekecuaan utama yang berlaku dalam algoritma ini.

Dalam Bab ini, saya akan membincangkan kekuatan, kelemahan dan masalah algoritma-algoritma yang timbul ketika melaksanakan proses dan simulasi bagi proses pengecaman aksara tulisan jawi dengan menggunakan bahasa perihalan perkakasan VHDL. Bab ini juga akan menyertakan cadangan bagi membentuk proses atau algoritma yang lebih mantap dan boleh menghasilkan output tanpa sebarang ralat dan permasalahan.

## **7.1 PERBINCANGAN BAGI ALGORITMA PEMBACAAN KOD RANTAIAN MELALUI KAEDAH TATASUSUNAN**

Seperti yang telah kita maklum, proses pengecaman aksara tulisan jawi ini dengan mengimplementasikan kaedah tatasusunan telah biasa didedahkan dalam bahasa pengaturcaraan seperti C, C++ dan Matlab. Semua ini lebih cenderung ke arah pengaturcaraan perisian. Namun, kajian yang saya usahakan ini adalah berdasarkan pengaturcaraan perkakasan. Saya telah cuba menggunakan kaedah tatasusunan dalam bahasa perihalan perkakasan VHDL bagi proses pembacaan kod rangkaian. Secara logiknya, kaedah ini boleh menghasilkan output yang diharapkan tanpa sebarang ralat dan sintak. Setelah diuji, terdapat masalah yang timbul iaitu kesilapan sintak pada kaedah tatasusunan itu sendiri. Seperti yang telah ditunjukkan dalam keputusan simulasi, output tidak dapat menghasilkan gabungan kod rangkaian. Jadi saya akan menerangkan bahagian dimana kesilapan utama yang dikesan dalam algoritma ini.



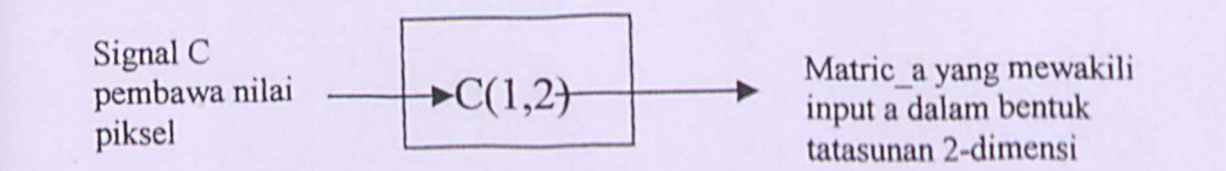
```
entity TITIKMULA is
port (
    -- WIZ BEGINPORTS (Entity Wizard command)

    a: in std_logic;
    clk: in std_logic;
    reset: in std_logic    ;
    value:out std_logic_vector(2 downto 0)

    -- WIZ ENDPORTS (Entity Wizard command)
);
end TITIKMULA;

type matric_a is array(0 to 15,15 downto 0) of bit;
signal c:matric_a;
```

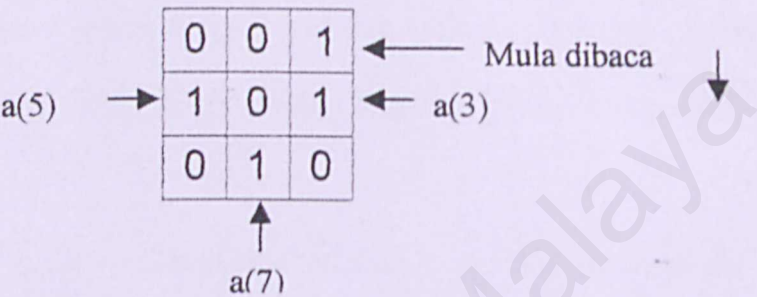
Sebagaimana diketahui, input yang diterima diwakili oleh a dan ini telah diisytiharkan pada kepala program. Selepas itu saya mengisytiharkan tatasusunan matric\_a sebagai medan pembacaan semua input yang diterima dalam bentuk matrik 4 X 4. Setiap piksel itu akan diwakili tatasusunan 2-dimensi. Signal C secara logik akan bertindak sebagai pembawa nilai-nilai piksel yang telah diwakili ini.



Oleh kerana pendekatan ini melanggar sintak kaedah tatasusunan, proses ini tidak dapat membaca input dan menghasilkan output sejak permulaan proses lagi. Kesimpulannya, keadah ini tidak dapat diteruskan kerana kesalahan sintak pada awal proses aturcara dan ini secara tidak langsung akan mempengaruhi keseluruhan perjalanan proses. Selain itu, oleh kerana kurang kemahiran pemahaman dan pengolahan sintak keadah tatasusunan ini adalah faktor utama proses ini tidak dapat dijalankan. Oleh yang demikian, secara logiknya kaedah tatasusunan ini adalah algoritma yang sesuai bagi proses pembacaan kod rantai ini. Saya akan menerangkan dengan lebih lanjut perlunya kaedah ini dalam bahagian kesimpulan dan cadangan nanti.

## 7.2 PERBINCANGAN BAGI ALGORITMA MEMBACA KOD RANTAIAN BERDASARKAN KEJIRANAN PIKSEL YANG TELAH DITETAPKAN

Algoritma ini dapat menghasilkan output gabungan kod rantaian seperti yang diharapkan. Tetapi masih terdapat kelemahan yang ada pada kaedah ini kerana pembacaan kod rantaian terhad. Ini kerana piksel-piksel tidak ditetapkan nilai sebelum proses dijalankan.



Sepertimana yang kita ketahui, pembacaan akan dijalankan pada satu piksel ke piksel yang berjiranan dari kanan ke kiri. Oleh kerana nilai pada piksel tidak disimpan atau ditetapkan nilainya maka had pembacaan hanya kepada 2 piksel berdekatan dan seterusnya. Contohnya seperti rajah di atas, kod rantaian bagi a(3) kepada a(7) boleh dibaca kerana berjiranan dan seterusnya, tetapi kod rantaian bagi a(7) kepada a(5) tidak dapat dibaca kerana walaupun berjiranan tetapi tidak seterusnya. Maksudnya disini a(5) hanya dapat dibaca daripada piksel a(1), a(2) dan a(4) sahaja. Ini kerana tiada penyimpanan nilai pada piksel-piksel tersebut sebelum proses pembacaan dijalankan.



7.3 Kesimpulannya, algoritma ini telah hampir memenuhi matlamat penghasilan output gabungan kod rangkaian itu tetapi disebabkan tiada penyimpanan nilai imej pada piksel dalam proses pembacaan adalah faktor utama algoritma ini tidak dapat diimplementasikan secara berkesan. Untuk penyimpanan nilai pada setiap piksel ini, kita perlu menggunakan 'buffer' terlebih dahulu dan ini akan dibincangkan dalam bahagian kesimpulan dan cadangan.

bagi algoritma kesimpulannya piksel yang ditetapkan pada kesimpulannya terutamanya penyimpanan nilai input pada piksel tersebut sebelum proses mengira kod rangkaian dijalankan.

Jadi di sini saya dapat mengidentifikasi faktor utama kesimpulannya pada kedua-dua algoritma ini:

- 1) Kesalahan dalam pengolahan dan peraturan simbol kod rangkaian
- 2) Tiada fungsi buffer yang boleh menyimpan nilai pada piksel bagi setiap kesimpulannya input

### 7.3.1 Kesalahan dalam pengolahan dan peraturan simbol kod rangkaian

Pada algoritma kod rangkaian yang saya gunakan ini tidak mempunyai ciri-ciri pengolahan kod rangkaian yang sebenar. Bagi proses pengolahan kod rangkaian jawi ini, kod rangkaian adalah kod yang paling awal diimplementasikan kerana proses pengolahan kod rangkaian pada piksel lebih mudah dilakukan. Ini kerana pengolahan adalah bermula pada kod rangkaian.

### 7.3 Kesimpulan Dan Cadangan

Setelah semua proses perancangan dan pengujian dijalankan bagi merealisasikan proses pengecaman aksara jawi ini, terdapat kelemahan yang timbul pada setiap algoritma yang saya gunakan. Kesilapan yang utama adalah lebih kepada sintak atau pun kesalahan pengolahan pengaturcaraan tersebut. Bagi algoritma kaedah tatasusunan, kesilapan utama adalah kesalahan dalam pengisytiharan tatasusunan itu sendiri manakala bagi algoritma kejiranan piksel yang ditetapkan pula kesalahannya tertumpu kepada penyimpanan nilai input pada piksel tersebut sebelum proses mengecam kod rangkaian dijalankan.

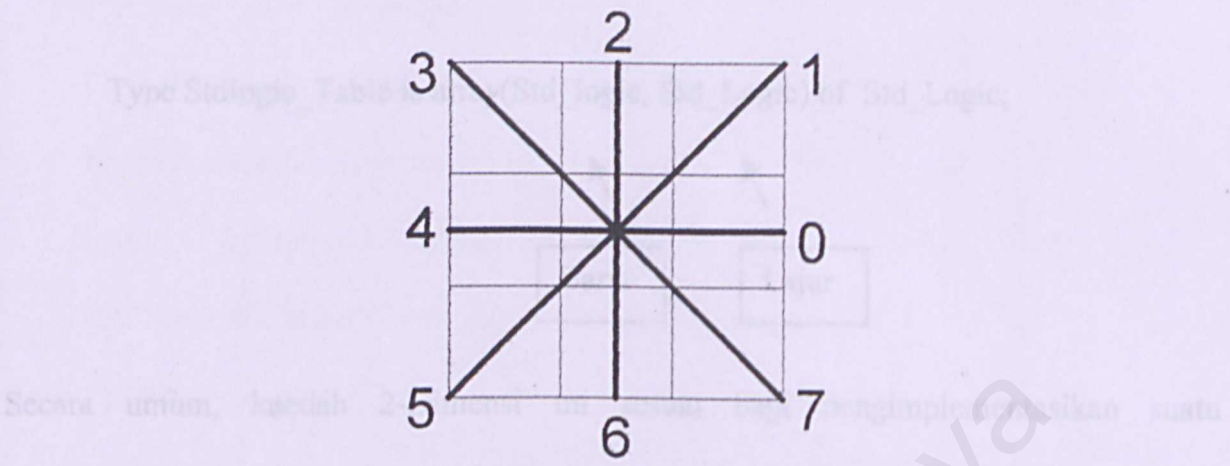
Jadi di sini saya dapat menklasifikasikan faktor utama kelemahan pada kedua-dua algoritma iaitu :

- 1) Kesalahan dalam pengolahan dan peraturan sintak kaedah tatasusunan
- 2) Tiada fungsi *buffer* yang boleh menyimpan nilai pada piksel bagi setiap kemasukan input

#### 7.3.1 Kesalahan dalam pengolahan dan peraturan sintak kaedah tatasusunan

Pada algoritma kaedah tatasusunan yang saya gunakan ini tidak menepati ciri-ciri pengolahan aturcara tatasusunan yang sebenar. Bagi proses pengecaman aksara tulisan jawi ini, kaedah tatasusunan adalah kaedah yang paling sesuai dimplementasikan kerana proses pembacaan kod rangkaian pada piksel lebih mudah dilakukan. Ini kerana pembacaan adalah bertumpuan kepada 8 arah.

Kaedah tatasusunan ini akan mewujudkan satu matrik yang mewakili 2 dimensi nilai. Contoh seperti di bawah.



Rajah 7.3.2 : Implementasi kaedah matrik tatasusunan bagi pembacaan arah kod rantaian

Merujuk pada rajah di atas, kita dapat meyakini bawah kaedah pembacaan matrik ini dapat menentukan arah kod rantaian dengan lebih berkesan. Saya akan menerangkan kaedah tatasusunan Multi-Dimensi secara ringkas.

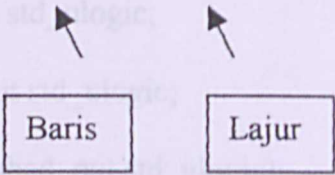
7.3.3 Tatasusunan Multi-Dimensi

Tatasusunan Multi-Dimensi adalah tatasusunan lebih kepada satu indek. Sebagai contoh, 2 objek berjenis “chaincode\_ Typ is *array(4 downto 0, 3 downto 0) of Std\_Logi*.;” mewakili 2 dimensi tatasusunan (X,Y), dimana satu indek mempunyai lima kemasukan dan satu indek mempunyai empat kemasukan. Objek ini akan bertindak seperti ‘checkerboard’ kepada 24 elemen itu.



7.3.4 Setiap elemen mewakili satu bit (of type Std\_Logic). Tatasusunan 2-Dimensi mengalamatkan sebagai baris dan lajur seterusnya seperti berikut:

Type Stdlogic\_Table is array(Std\_logic, Std\_Logic) of Std\_Logic;



Secara umum, kaedah 2-Dimensi ini sesuai bagi pengimplementasikan suatu pemalar(constant) dan ‘lookup table’ bagi menyimpan nilai kemasukan input. Pendekatan ini adalah paling sesuai dan berkesan dalam pembacaan arah kod rantai.

7.3.4 Tiada fungsi *buffer* yang boleh meyimpan nilai pada piksel bagi setiap kemasukan input

Dalam algoritma kedua iaitu kaedah pembacaan berdasarkan kejiranan piksel mempunyai kelemahan pada segi penyimpanan nilai pada piksel bagi setiap kemasukan input. Apa yang telah saya lakukan pada algoritma ini adalah melaksanakan proses pembacaan pada piksel tanpa menyimpan terlebih nilai input pada piksel itu sendiri. Jadi, logiknya proses tidak dapat membaca nilai piksel secara tepat.

Oleh yang demikian, langkah yang perlu diambil adalah dengan melaksanakan dahulu fungsi ‘buffer’ bagi menyimpan dan menetapkan nilai kemasukan input. Jadi, apabila piksel-piksel menerima input, ia akan dapat menyimpan nilai itu ke dalam ‘buffer’.

### 7.3.4 Contoh penggunaan buffer secara ringkas

Entity CACHE Is

Port (Clk, Reset: in std\_ulogic;

CacheHit, CacheMiss, Rd, Wr: in std\_ulogic;

MemFinished, CacheFinished: in std\_ulogic;

RdMem, WrMem, AbortMem: out std\_ulogic;

RdCache, WrCache, Check, Finished: out std\_ulogic);

End CACHE;

Architecture CACHE\_A of CACHE Is

signal current\_state: state;

Begin

process(Reset, Clk)

begin

if Reset = '1' then

current\_state <= RESET\_STATE; -- Important note: we must

RdMem <= '0'; WrMem <= '0'; -- include all registered

AbortMem <= '0'; -- outputs here, even if they

RdCache <= '0'; WrCache <= '0';-- do not all require a

-- a reset. If any are left

Check <= '0'; -- out, we will have unexpected

Finished <= '0'; -- errors and/or incorrect

```
elsif rising_edge(Clk) then    -- synthesis results.
```

```
-- Default values for outputs...
```

```
RdMem <= '0'; WrMem <= '0';
```

```
AbortMem <= '0';
```

```
RdCache <= '0'; WrCache <= '0';
```

```
Check <= '0'; Finished <= '0';
```

Pada aturcara 'Cache' di atas telah membina buffer bagi penyimpanan dan penetapan nilai untuk proses input dan output tersebut.

### 7.3.5 Kesimpulan

Setelah semua bab iaitu Pembangunan Sistem, Pengujian Sistem dan Perbincangan dilakukan, saya mendapati kaedah terbaik bagi membuatkan proses pengecaman aksara tulisan jawi yang mampu menghasilkan output yang diharapkan adalah dengan :

- 1) Membina terlebih dahulu fungsi buffer bagi penyimpanan nilai input pada piksel sebelum proses dijalankan.
- 2) Menggunakan pendekatan kaedah tatasusunan Multi-Dimensi yang betul.

Dengan pendekatan ini , saya yakin masalah yang timbul pada kedua-dua algoritma tersebut dapat diatasi dan dapat menghasilkan ouput gabungan kod rantai dengan lebih berkesan.

*wassalam*



## SENARAI RUJUKAN

1. Omar bin Abdul Rahim. Pengecaman Aksara Tulisan Jawi, Latihan Ilmiah Universiti Malaya.
2. Castleman ,Kenneth R. Digital Image Processing . Prentice Hall International Edition. 1996
3. Sonka Milan, Hlavac Vaclav, Boyle Roger. Image Processing, Analysis, and Machine Vision.Second Edition. PWS Publishing.
4. Bhasker, J. VHDL Primer. Third Edition Bell Laboratories, Lucent Technologies Allentown, PA. Prentice Hall PTR.
5. Rushton, Andrew VHDL for Logic Synthesis. Second Edition. TransEDA Limited, Southamton, UK. John Wiley & Sons. 2001.
6. Cohen, Ben. VHDL Answers to Frequently Asked Questions. Kluwer Academic Publishers Boston/Dordrecht/London.
7. Zainalabedin Navabi. VHDL Analysis And Modelling Of Digital Systems.Second Edition McGraw-Hill.
8. Naylor, David. and Jones, Simon. VHDL : A Logic Synthesis Approach. Chapman & Hall. 1997
9. Chain Code, [http://ltswww.epfl.ch/pub\\_files/brigger/thesis\\_html/node14.html](http://ltswww.epfl.ch/pub_files/brigger/thesis_html/node14.html)